

```
1 /*<html>
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-0.6.5--2020-10-10--SatoxITS</span>
7 <title id="GshTitle">GShell-0.6.5 by SatoxITS</title>
8
9 <div id="GshTopbar" class="MetaWindow"></div>
10 <div id="GshPerfMon"></div>
11 <div id="GshSidebar" class="SbSidebar"><div id="GshIndexer"></div></div>
12 <div id="GshMain">
13 <header id="GshBanner" height="100px" onclick="shiftBG();">
14 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.6.5 // 2020-10-10 // SatoxITS</note></div>
15 </header>
16
17 //<!-- ----- Work { ----- -->
18 //<span id="Topbar_WorkCodeSpan">
19 /*
20 <details><summary>Topbar</summary>
21 <!-- ----- Topbar // 2020-1008 SatoxITS { -->
22 <h2>Topbar</h2>
23 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
24 <input id="Topbar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
25 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
26 <span id="Topbar_WorkCodeView"></span>
27 </details>
28
29 <style>
30 .ConfigIcon {
31   position:absolute;
32   top:-6px;
33   left:92%;
34   width:32px;
35   height:32px;
36 }
37 .MetaWindow {
38   z-index:1000;
39   position:relative;
40   display:block;
41   overflow:visible !important;
42   width:99.9%;
43   height:22px;
44   top:-22px;
45   border:1px solid #22a;
46   margin:0px;
47   left:0.0%;
48   line-height:1.0;
49   font-family:Georgia;
50   color:#fff;
51   font-size:12pt;
52   text-align:center;
53   vertical-align:middle;
54   padding:4px;
55   xxbackground-color:rgba(0,8,170,0.8);
56   background-color:#3a4861;xxx-PBlue;
57   vertical-align:middle;
58 }
59 .MetaWindow:hover {
60   color:#000;
61   border:1px solid #22a;
62   background-color:rgba(255,255,255,1.0);
63 }
64 </style>
65 <script>
66 function Topbar_openWorkCodeView(){
67   function Topbar_showWorkCode(){
68     showHtmlCode(Topbar_WorkCodeView,Topbar_WorkCodeSpan);
69   }
70   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
71 }
72 Topbar_openWorkCodeView();
73 function Gshell_initTopbar(){
74   GshTopbar.innerHTML = GshTitle.innerHTML;
75   //<img id="ConfigIcon" class="ConfigIcon">
76   if( true ){
77     cfgi = document.createElement('img');
78     cfgi.id = 'ConfigIcon';
79     cfgi.setAttribute( 'class', 'ConfigIcon' );
80     GshTopbar.appendChild(cfgi);
81     cfgi.src = ConfigICON_DATA;
82   }
83 }
84 </script>
85 <!-- Topbar_WorkCodeSpan } -->
86 /* //</span>
87 //<!-- ----- Work } ----- -->
88
89
90 //<!-- ----- Work { ----- -->
91 //<span id="Indexer_WorkCodeSpan">
92 /*
93 <details><summary>Indexer</summary>
94 <!-- ----- Indexer // 2020-1007 SatoxITS { -->
95 <h2>Indexer</h2>
96 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
97 <input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
98 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
99 <span id="Indexer_WorkCodeView"></span>
100 </details>
101 <style id="SidebarIndex">
102 #gsh {
103   display:block;
104   xxxoverflow:scroll !important;
105 }
106 #GshMain {
107   position:relative;
108   width:80% !important;
109   left:19.5% !important;
110 }
111 #GshSidebar {
112   z-index:0;
113   position:relative !important;
114   overflow:auto;
115   resize:both !important;
116   xxoverflow-y:hidden !important;
117   xxheight:100px !important;
118   xxdisplay:inline !important;
119   left:0px;
120   top:0px;
121   width:19.5%;
122   min-width:80px;
123   xxheight:100% !important;
124   height:0px;
```

```

125 color:#f00;
126 xxbackground-color:rgba(64,64,64,0.5);
127 xxbackground-color:#DFE3EB;xxx-PBlue;
128 background-color:#eeeeee;xxx-PBlue;
129 }
130 #GshPerfMon {
131 position:relative;
132 z-index:10000000;
133 xxheight:12pt;
134 font-family:monospace, Courier New !important;
135 font-size:9pt !important;
136 color:#f84;
137 top:-20px;
138 }
139 #GshSidebar:hover {
140 z-index:10000000;
141 overflow-x:visible !important;
142 background-color:rgba(255,255,255,0.7);
143 width:508;
144 }
145 #GshIndexer {
146 z-index:0;
147 position:relative;
148 resize:both !important;
149 height:100%;
150 left:0px;
151 top:0px;
152 scroll-behavior: overflow !important;
153 padding-left:4pt;
154 font-size:0.5em;
155 white-space:nowrap;
156 xxx-background-color:rgba(64,160,64,0.6) !important;
157 color:#7794c6;xxx-PBlue;
158 xxbackground-color:#DFE3EB;xxx-PBlue;
159 background-color:#eeeeee;xxx-PBlue;
160 }
161 #GshIndexer:hover {
162 z-index:10000000;
163 overflow-x:visible !important;
164 color:#000000 !important;xxx-PBlue;
165 xxbackground-color:#FFFFFF;xxx-PBlue;
166 background-color:rgba(255,255,255,0.7);
167 padding-right:0px;
168 width:80%;
169 }
170 #GshIndexer:select {
171 color:#000000 !important;xxx-PBlue;
172 background-color:#FFFFFF;xxx-PBlue;
173 }
174 .IndexLine {
175 font-size:8pt !important;
176 font-family:Georgia;
177 display:block;
178 xxx-color:#fff;
179 xxx-color:#fffff5;xxx-PBlue;
180 xxx-color:#41516d;xxx-PBlue;
181 xxx-color:#7794c6;xxx-PBlue;
182 padding-right:4pt;
183 }
184 .IndexLine:hover {
185 font-size:10pt!important;
186 xxx-color:#228;
187 xxx-background-color:#fff;
188 xxcolor:#fff;xxx-PBlue;
189 color:#516487;xxx-PBlue;
190 background-color:rgba(220,220,255,1.0);xxx-PBlue;
191 xxtext-shadow:1px 1px #f3f;
192 text-shadow:1px 1px #eee;
193 xxbackground-color:#516487;xxx-PBlue;
194 xxtext-decoration:underline !important;
195 }
196 </style>
197
198 <script id="Indexer_WorkScript">
199 function Indexer_OpenWorkCodeView(){
200     function Indexer_showWorkCode(){
201         showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
202     }
203     Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
204 }
205 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
206 Indexer_openWorkCodeView();
207
208 var startPerfDate = new Date();
209 var prevPerfDate = startPerfDate;
210 function ShowResourceUsage(){
211     d = new Date();
212     perf = '';
213     perf += '<'+font color="gray">UA:' + window.navigator.userAgent + '<'+/font><br>\n';
214     perf += DateShort0(startPerfDate) + '<br>\n';
215     perf += DateShort() + '<br>\n';
216     elps = d.getTime() - startPerfDate.getTime();
217     itvl = d.getTime() - prevPerfDate.getTime();
218     perf += 'Elapsed: ' + elps/1000 + ' s<br>\n';
219     perf += 'Skew: ' + (itvl-1000) + ' ms<br>\n';
220     prevPerfDate = d;
221
222     if( performance.memory !== undefined ){
223         m0 = performance.memory;
224         mu0 = (m0.usedJSHeapSize / 1000000).toFixed(6);
225         perf += 'Memory: '+mu0+' MB<br>\n';
226     }
227     perf += '<br>\n';
228
229 //GshSidebar.innerHTML = perf;
230 GshPerfMon.innerHTML = perf;
231 //GshIndexer.innerHTML = 'Memory: '+mu0+' MB';
232 //console.log('-- PerfMon heap: '+mu0+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
233 if( true ){
234     GshSidebar.style.zIndex = 1000;
235     GshIndexer.style.zIndex = 0;
236     GshPerfMon.style.zIndex = 1;
237     //GshSidebar.appendChild(GshPerfMon);
238     if( document.getElementById('primary') == null ){ // not in WordPress
239         GshPerfMon.style.position = 'absolute';
240     }
241     GshPerfMon.style.display = 'block';
242     GshPerfMon.style.marginLeft = '4px';
243     GshPerfMon.style.top = '45px';
244     GshPerfMon.style.left = '0px';
245 }
246 var iserno = 0;

```

```

249 var GeneratedId = 0;
250 function generateIndex(ni,e,chv,nch,ht){
251   // https://developer.mozilla.org/en-US/docs/Web/API/Element
252   c = '';
253   if( e.classList != null ){
254     c = e.classList.value;
255   }
256   //console.log('-- <' +e.nodeName+ '> #' +e.id+ ' .'+c+ ' '+e.attributes);
257   if( e.nodeName == '#text' ) return '';
258   if( e.nodeName == '#comment' ){ return ''; }
259   if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
260     id = e.innerHTML;
261     GeneratedId += 1;
262     eid = 'GeneratedId-' +GeneratedId;
263     e.id = eid;
264   }else{
265     if( e.nodeName == 'SUMMARY' ){
266       id = e.innerHTML;
267       GeneratedId += 1;
268       eid = 'Generatedid-' +GeneratedId;
269       e.id = eid;
270     }else{
271       if( (and(e.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxxxxxentry-content') ){
272         console.log('<!-- DIV entry-content begin');
273         id = e.innerHTML;
274         GeneratedId += 1;
275         eid = 'GeneratedId-' +GeneratedId;
276         e.id = eid;
277         console.log('<!-- DIV entry-content end hash-child=' ,e.hasChildNodes());
278       }else{
279         if( e.id == '' || e.id == 'undefined' ){
280           return '';
281         }else{
282           id = '#'+e.id;
283           eid = e.id;
284         }
285       iserno += 1;
286       ht = '&lt;+' +div id="GeneratedRef_'+iserno+'" class="IndexLine" href="#"&gt;';
287       + iserno+' '+ni+':'+e.nodeName + ':' + id;
288       if( e.id == '' || e.id == 'undefined' ) return ht + '&lt;+'/div&gt;';
289       if( !e.hasChildNodes() ) return ht + '&lt;+'/div&gt;';
290       chv = e.childNodes;
291       nch = e.childNodes.length;
292       if( chv != null ) nch = chv.length;
293       ht += ' ('+nch+')' + '&lt;+'/div&gt;';
294       for( let i = 0; i &lt; chv.length; i++ ){
295         sec = ni+'.';
296         if( ni == '' ) sec = i;
297         ht += generateIndex(sec,chv[i],null,0);
298       }
299       return ht;
300     }
301   function onClickIndex(e){
302     tid = e.target.id;
303     tge = document.getElementById(tid);
304     eid = tge.getAttribute('href');
305     rx = tge.getBoundingClientRect().left.toFixed(0)
306     ry = tge.getBoundingClientRect().top.toFixed(0)
307     if( false ){
308       alert('index clicked mouse(x=' +e.x+', y=' +e.y+')'
309             + 'ntid=' + tid + ' rx=' +rx + ',ry=' +ry
310             + '\neid=' + eid + '\n'
311             + '\nhtml=' + tge.outerHTML);
312     }
313     ee = document.getElementById(eid);
314     sx = 'NaN';
315     sy = ee.getBoundingClientRect().top;
316     console.log('sx=' +sx+',sy=' +sy);
317     ee.scrollIntoView();
318     window.scrollTo(sx,sy)
319     //window.scrollTo(left:'Non',top:sy,behavior:'smooth'});
320   }
321   function Indexer_afterLoaded(){
322     sideindex = document.getElementById('GshIndexer');
323     ht = '&lt;+' +h3+&gt;-Index'+ '/h3';
324     ht += generateIndex("",document.getElementById('gsh'),null,0,'');
325     if( (pri = document.getElementById('primary')) != null ){
326       ht += generateIndex("",pri,null,0,'');
327     }
328     ht += '&lt;+' +br+';
329     ht += '&lt;+' +br+';
330     ht += '&lt;+' +br+';
331     ht += '&lt;+' +br+';
332     sideindex.innerHTML = ht;
333     sideindex.addEventListener('click',onClickIndex);
334     if( (pri = document.getElementById('primary')) != null ){
335       console.log('-- Seems in WordPress');
336       pri.style.zIndex = 2000;
337       GshSidebar.style.setProperty('position','relative','important');
338       GshSidebar.style.top = '-1400px';
339       //GshSidebar.style.setProperty('position','absolute','important');
340       //GshSidebar.style.top = '0px';
341       GshSidebar.style.setProperty('width','200px','important');
342       GshSidebar.style.setProperty('overflow','scroll','important');
343       GshSidebar.style.resize = 'both';
344       GshSidebar.style.left = '-100px';
345       GshIndexer.style.left = '100px';
346       GshIndexer.style.height = '1400px';
347       gsh.appendChild(GshSidebar); // change parent
348     }else{
349       console.log('-- Seems not in WordPress');
350       GshSidebar.style.setProperty('position','fixed','important');
351     }
352   }
353   //document.addEventListener('load',Indexer_afterLoaded);
354
355 DestroyIndexBar = function(){
356   sideindex = document.getElementById('GshIndexer');
357   sideindex.innerHTML = "";
358 }
359 &lt;/script&gt;
360
361 &lt;!-- Indexer_WorkCodeSpan } --&gt;
362 /* */ /&lt;/span&gt;
363 //&lt;!-- ----- Work } ----- --&gt;
364
365 /*
366 &lt;h2&gt;GShell // a General purpose Shell built on the top of Golang&lt;/h2&gt;
367 &lt;p&gt;
</pre>

```

```

373 <note>
374 It is a shell for myself, by myself, of myself. --SatoxITS(^-^)
375 <a href="#">gsh-0.6.2.go.html>prev.</a>
376 </note>
377 </p>
378 <div id="GJFactory_x"></div>
379
380 <div>
381 <span id="GshMenu">
382 <span class="GshMenuL" id="GshMenuEdit" onclick="html_edit();">Edit</span>
383 <span class="GshMenuL" id="GshMenuSave" onclick="html_save();">Save</span>
384 <span class="GshMenuL" id="GshMenuLoad" onclick="html_load();">Load</span>
385 <span class="GshMenuL" id="GshMenuVers" onclick="html_ver0();">Vers</span>
386 <span id="gsh-Wnid" onclick="win_jump('0.1');">0</span>
387 <span class="GshMenuL" id="gsh-menu-exit" onclick="html_close();">Exit</span>
388 <span class="GshMenuL" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
389 <span class="GshMenuL" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
390 <span class="GshMenuL" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
391 <span class="GshMenuL" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
392 <span class="GshMenuL" id="GshMenuSign" onclick="html_sign(this); style="">Source</span>
393 <!-- / <span id="gsh-menu-pure" onclick="html_pure(this);>Pure</span> -->
394 </span>
395 </div>
396 /*
397 */
398 /*
399 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
400 <h3>Fun to create a shell</h3>
401 <p>For a programmer, it must be far easy and fun to create his own simple shell
402 rightly fitting to his favor and necessities, than learning existing shells with
403 complex full features that he never use.
404 I, as one of programmers, am writing this tiny shell for my own real needs,
405 totally from scratch, with fun.
406 </p><p>
407 For a programmer, it is fun to learn new computer languages. For long years before
408 writing this software, I had been specialized to C and early HTML2 :).
409 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
410 on demand as a novice of these, with fun.
411 </p><p>
412 This single file "gsh.go", that is executable by Go, contains all of the code written
413 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
414 HTML file that works as the viewer of the code of itself, and as the "home page" of
415 this software.
416 </p><p>
417 Because this HTML file is a Go program, you may run it as a real shell program
418 on your computer.
419 But you must be aware that this program is written under situation like above.
420 Needless to say, there is no warranty for this program in any means.
421 </p>
422 <address>Aug 2020, SatoxITS (sato@its-more.jp)</address>
423 </details>
424 /*
425 */
426 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
427 </p>
428 <h3>Cross-browser communication</h3>
429 <p>
430 ... to be written ...
431 </p>
432 <h3>Vi compatible command line editor</h3>
433 <p>
434 The command line of GShell can be edited with commands compatible with
435 <a href="#">https://www.washington.edu/computing/unix/vi.html>vi</a>.
436 As in vi, you can enter <i><b>command mode</b></i> by <b>ESC</b> key,
437 then move around in the history by <b><code>j k / ? n N</code></b>,
438 or within the current line by <b><code>l h f w b 0 $ %</code></b> or so.
439 </p>
440 </details>
441 /*
442 */
443 <details id="gsh-gindex">
444 <summary>Index</summary><div class="gsh-src">
445 Documents
446 <span class="gsh-link" onclick="jumpto_JavaScriptView();">Command summary</span>
447 Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
448 Package structures
449 <a href="#">#import>import</a>
450 <a href="#">#struct>struct</a>
451 Main functions
452 <a href="#">#comexpansion>str-expansion</a> // macro processor
453 <a href="#">#finder>finder</a> // builtin find + du
454 <a href="#">#grep>grep</a> // builtin grep + wc + cksum +
455 <a href="#">#plugin>plugin</a> // plugin commands
456 <a href="#">#exec-commands>system</a> // external commands
457 <a href="#">#builtin>builtin</a> // builtin commands
458 <a href="#">#network>network</a> // socket handler
459 <a href="#">#remote-sh>remote-sh</a> // remote shell
460 <a href="#">#redirect>redirect</a> // Stdin/Out redireciton
461 <a href="#">#history>history</a> // command history
462 <a href="#">#usage>usage</a> // resource usage
463 <a href="#">#encode>encode</a> // encode / decode
464 <a href="#">#IME>IME</a> // command line IME
465 <a href="#">#getline>getline</a> // line editor
466 <a href="#">#scanf>scanf</a> // string decomposer
467 <a href="#">#interpreter>interpreter</a> // command interpreter
468 <a href="#">#main>main</a>
469 </span>
470 JavaScript part
471 <a href="#">#script-src-view>script-src-view</a>
472 <a href="#">#gsh-data-frame>gsh-data-frame</a>
473 CSS part
474 <a href="#">#style-src-view>style-src-view</a>
475 References
476 <a href="#" class="gsh-link" onclick="jumpto_WholeView();">Internal</a>
477 <a href="#">#gsh-reference>gsh-reference</a>
478 Whole parts
479 <a href="#">#whole-src-view>whole-src-view</a>
480 <a href="#">#whole-src-view>whole-src-view</a>
481 <a href="#">#whole-src-view>whole-src-view</a>
482 </div>
483 </details>
484 /*
485 */
486 //<details id="gsh-gocode">
487 //<summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
488 // gsh - Go lang based Shell
489 // (c) 2020 ITS more Co., Ltd.
490 // 2020-0807 created by SatoxITS (sato@its-more.jp)
491
492 package main // gsh main
493
494 // <a name="#">import>Imported packages</a> // <a href="#">https://golang.org/pkg/>Packages</a>
495 import (
496   "fmt" // <a href="#">https://golang.org/pkg/fmt/>fmt</a>

```

```

497   "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
498   "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
499   "sort" // <a href="https://golang.org/pkg/sort/">sort</a>
500   "time" // <a href="https://golang.org/pkg/time/">time</a>
501   "bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
502   "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
503   "os" // <a href="https://golang.org/pkg/os/">os</a>
504   "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
505   "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
506   "net" // <a href="https://golang.org/pkg/net/">net</a>
507   "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
508   // "html" // <a href="https://golang.org/pkg/html/">html</a>
509   "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
510   "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
511   "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
512   "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
513   "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
514   // "gshdata" // gshell's logo and source code
515   "hash/crc32" // <a href="https://golang.org/pkg/unicode/hash/crc32/">crc32</a>
516   "golang.org/x/net/websocket"
517 }
518
519 // // 2020-0906 added,
520 // // <a href="https://golang.org/cmd/cgo/">CGo</a>
521 // #include "poll.h" // <poll.h> to be closed as HTML tag :-p
522 // typedef struct { struct pollfd fdv[8]; } pollFdV;
523 // int pollx(pollFdV *fdv, int nfds, int timeout){
524 //     return poll(fdv->fdv,nfds,timeout);
525 // }
526 import "C"
527
528 // // 2020-0906 added,
529 func CFpollInl(fp*os.File, timeoutUs int)(ready uintptr){
530     var fdv = C.pollFdV{}
531     var nfds = 1
532     var timeout = timeoutUs/1000
533
534     fdv.fd[0].fd = C.int(fp.Fd())
535     fdv.fd[0].events = C.POLLIN
536     if( 0 < EventRecvFd ){
537         fdv.fd[1].fd = C.int(EventRecvFd)
538         fdv.fd[1].events = C.POLLIN
539         nfds += 1
540     }
541     r := C.pollx(&fdv,C.int(nfds),C.int(timeout))
542     if( r <= 0 ){
543         return 0
544     }
545     if(int(fdv.fd[1].revents) & int(C.POLLIN)) != 0 {
546         //fprintf(stderr,"--De-- got Event\n");
547         return uintptr(EventFdOffset + fdv.fd[1].fd)
548     }
549     if(int(fdv.fd[0].revents) & int(C.POLLIN)) != 0 {
550         return uintptr(NormalFdOffset + fdv.fd[0].fd)
551     }
552     return 0
553 }
554
555 const (
556     NAME = "gsh"
557     VERSION = "0.6.5"
558     DATE = "2020-10-10"
559     AUTHOR = "SatoxITS(^~)//"
560 )
561 var (
562     GSH_HOME = ".gsh" // under home directory
563     GSH_PORT = 9999
564     MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
565     PROMPT = "> "
566     LINESIZE = (8*1024)
567     PATHSEP = ":" // should be ";" in Windows
568     DIRSEP = "/" // canbe \' in Windows
569 )
570
571 // -xx logging control
572 // --A-- all
573 // --I-- info.
574 // --D-- debug
575 // --T-- time and resource usage
576 // --W-- warning
577 // --E-- error
578 // --F-- fatal error
579 // --Xn-- network
580
581 // <a name="struct">Structures</a>
582 type GCommandHistory struct {
583     StartAt    time.Time // command line execution started at
584     EndAt     time.Time // command line execution ended at
585     ResCode    int       // exit code of (external command)
586     CmdError   error    // error string
587     OutData   *os.File // output of the command
588     Foundfile []string  // output - result of ufind
589     Rusageev [2]syscall.Rusage // Resource consumption, CPU time or so
590     Cmdid     int       // maybe with identified with arguments or impact
591     // redirecton commands should not be the Cmdid
592     WorkDir   string   // working directory at start
593     WorkDirX  int       // index in ChdirHistory
594     CmdLine   string   // command line
595 }
596 type GChdirHistory struct {
597     Dir      string
598     MovedAt  time.Time
599     CmdIndex int
600 }
601 type CmdMode struct {
602     BackGround bool
603 }
604 type Event struct {
605     when      time.Time
606     event     int
607     evarg    int64
608     CmdIndex int
609 }
610 var CmdIndex int
611 var Events []Event
612 type PluginInfo struct {
613     Spec      *plugin.Plugin
614     Addr      plugin.Symbol
615     Name      string // maybe relative
616     Path      string // this is in Plugin but hidden
617 }
618 type GServer struct {
619     host      string
620     port      string

```

```

621 }
622 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
623 const ( // SumType
624     SUM_ITEMS = 0x000001 // items count
625     SUM_SIZE = 0x000002 // data length (simply added)
626     SUM_SIZEHASH = 0x000004 // data length (hashed sequence)
627     SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
628     // also envelope attributes like time stamp can be a part of digest
629     // hashed value of sizes or mod-date of files will be useful to detect changes
630
631     SUM_WORDS = 0x000010 // word count is a kind of digest
632     SUM_LINES = 0x000020 // line count is a kind of digest
633     SUM_SUM64 = 0x000040 // simple add of bytes, useful for human too
634
635     SUM_SUM32_BITS = 0x000100 // the number of true bits
636     SUM_SUM32_2BYTE = 0x000200 // 16bits words
637     SUM_SUM32_4BYTE = 0x000400 // 32bits words
638     SUM_SUM32_8BYTE = 0x000800 // 64bits words
639
640     SUM_SUM16_BSD = 0x001000 // UNIXsum -sum -bsd
641     SUM_SUM16_SYSV = 0x002000 // UNIXsum -sum -sysv
642     SUM_UNIXFILE = 0x004000
643     SUM_CRCIEEE = 0x008000
644 )
645 type CheckSum struct {
646     Files int64 // the number of files (or data)
647     Size int64 // content size
648     Words int64 // word count
649     Lines int64 // line count
650     SumType int
651     Sum64 uint64
652     Crc32Table crc32.Table
653     Crc32Val uint32
654     Sum16 int
655     Ctime time.Time
656     Atime time.Time
657     Mtime time.Time
658     Start time.Time
659     Done time.Time
660     RusgAtStart [2]syscall.Rusage
661     RusgAtEnd [2]syscall.Rusage
662 }
663 type ValueStack [][]string
664 type GshContext struct {
665     StartDir string // the current directory at the start
666     Getline string // gsh-getline command as a input line editor
667     ChdirHistory []GchdirHistory // the 1st entry is wd at the start
668     gshPA syscall.ProcAttr
669     CommandHistory []GCommandHistory
670     CmdCurrent GCommandHistory
671     BackGround bool
672     BackGroundJobs []int
673     LastRusage syscall.Rusage
674     GshHomeDir string
675     TerminalId int
676     CmdTrace bool // should be [map]
677     CmdTime bool // should be [map]
678     PluginFuncs []PluginInfo
679     iValues []string
680     iDelimiter string // field separator of print out
681     iFormat string // default print format (of integer)
682     iValStack ValueStack
683     LastServer GServer
684     RSERV string // [gsh://]host[:port]
685     RWD string // remote (target, there) working directory
686     lastCheckSum CheckSum
687 }
688 }
689 func nsleep(ns time.Duration){
690     time.Sleep(ns)
691 }
692 func usleep(ns time.Duration){
693     nsleep(ns*1000)
694 }
695 func msleep(ns time.Duration){
696     nsleep(ns*1000000)
697 }
698 func sleep(ns time.Duration){
699     nsleep(ns*1000000000)
700 }
701 }
702 func strBegins(str, pat string)(bool{
703     if len(pat) <= len(str){
704         yes := str[0:len(pat)] == pat
705         //fmt.Printf("-D-- strBegins(%v,%v)=%v\n",str,pat,yes)
706         return yes
707     }
708     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
709     return false
710 }
711 }
712 func isin(what string, list []string) bool {
713     for _, v := range list {
714         if v == what {
715             return true
716         }
717     }
718     return false
719 }
720 func isinX(what string,list[]string)(int){
721     for i,v := range list {
722         if v == what {
723             return i
724         }
725     }
726     return -1
727 }
728 func env(opts []string) {
729     env := os.Environ()
730     if isin("-s", opts){
731         sort.Slice(env, func(i,j int) bool {
732             return env[i] < env[j]
733         })
734     }
735     for _, v := range env {
736         fmt.Printf("%v\n",v)
737     }
738 }
739 }
740
741 // - rewriting should be context dependent
742 // - should postpone until the real point of evaluation
743 // - should rewrite only known notation of symbol
744 func scanInt(str string)(val int,leng int){

```

```

745     leng = -1
746     for i,ch := range str {
747         if '0' <= ch && ch <= '9' {
748             leng = i+1
749         }else{
750             break
751         }
752     }
753     if 0 < leng {
754         ival,_ := strconv.Atoi(str[0:leng])
755         return ival,leng
756     }else{
757         return 0,0
758     }
759 }
760 func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
761     if len(str[i+1:]) == 0 {
762         return 0,rstr
763     }
764     hi := 0
765     histlen := len(gshCtx.CommandHistory)
766     if str[i+1] == '!' {
767         hi = histlen - 1
768         leng = 1
769     }else{
770         hi,leng = scanInt(str[i+1:])
771         if leng == 0 {
772             return 0,rstr
773         }
774         if hi < 0 {
775             hi = histlen + hi
776         }
777     }
778     if 0 <= hi && hi < histlen {
779         var ext byte
780         if 1 < len(str[i+leng:]) {
781             ext = str[i+leng:][1]
782         }
783         //fmt.Printf("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
784         if ext == 'f' {
785             leng += 1
786             xlist := []string{}
787             list := gshCtx.CommandHistory[hi].FoundFile
788             for _,v := range list {
789                 //list[i] = escapeWhiteSP(v)
790                 xlist = append(xlist,escapeWhiteSP(v))
791             }
792             //rstr += strings.Join(list," ")
793             rstr += strings.Join(xlist," ")
794         }else
795         if ext == '*' || ext == 'd' {
796             // !N@ .. workdir at the start of the command
797             leng += 1
798             rstr += gshCtx.CommandHistory[hi].WorkDir
799         }else{
800             rstr += gshCtx.CommandHistory[hi].CmdLine
801         }
802     }else{
803         leng = 0
804     }
805     return leng,rstr
806 }
807 func escapeWhiteSP(str string)(string){
808     if len(str) == 0 {
809         return "\\\z" // empty, to be ignored
810     }
811     rstr := ""
812     for _,ch := range str {
813         switch ch {
814             case '\\': rstr += "\\\\\\""
815             case ' ': rstr += "\\s"
816             case '\t': rstr += "\\t"
817             case '\r': rstr += "\\r"
818             case '\n': rstr += "\\n"
819             default: rstr += string(ch)
820         }
821     }
822     return rstr
823 }
824 func unescapeWhiteSP(str string)(string){ // strip original escapes
825     rstr := ""
826     for i := 0; i < len(str); i++ {
827         ch := str[i]
828         if ch == '\\' {
829             if i+1 < len(str) {
830                 switch str[i+1] {
831                     case 'z':
832                         continue;
833                 }
834             }
835             rstr += string(ch)
836         }
837     }
838     return rstr
839 }
840 func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
841     ustrv := []string()
842     for _,v := range strv {
843         ustrv = append(ustrv,unescapeWhiteSP(v))
844     }
845     return ustrv
846 }
847
848 // <a name="comexpansion">str-expansion</a>
849 // - this should be a macro processor
850 func strsubst(gshCtx *GshContext,str string,histonly bool) string {
851     rbuff := []byte{}
852     if false {
853         //8U Unicode should be cared as a character
854         return str
855     }
856     //rstr := "" // escape characer mode
857     inEsc := 0
858     for i := 0; i < len(str); i++ {
859         //fmt.Printf("--D--Subst %v:%v\n",i,str[i:])
860         ch := str[i]
861         if inEsc == 0 {
862             if ch == '!' {
863                 //leng,xrstr := substHistory(gshCtx,str,i,rstr)
864                 leng,rs := substHistory(gshCtx,str,i,"")
865                 if 0 < leng {
866                     //_,rs := substHistory(gshCtx,str,i,"")
867                     rbuff = append(rbuff,[byte(rs)...])
868                     i += leng
869                 }
870             }
871         }
872     }
873     return string(rbuff)
874 }

```

```

869         //rstr = xrstr
870         continue
871     }
872     switch ch {
873     case '\\': inEsc = '\\'; continue
874     //case '%': inEsc = '%'; continue
875     case '$':
876     }
877 }
878 switch inEsc {
879 case '\\':
880 case '\\':
881     switch ch {
882     case '\\': ch = '\\'
883     case 's': ch = ' '
884     case 't': ch = '\t'
885     case 'r': ch = '\r'
886     case 'n': ch = '\n'
887     case 'z': inEsc = 0; continue // empty, to be ignored
888   }
889   inEsc = 0
890 case '\\':
891   switch {
892   case ch == '%': ch = '%'
893   case ch == 'T':
894     //rstr = rstr + time.Now().Format(time.Stamp)
895   rs := time.Now().Format(time.Stamp)
896   rbuff = append(rbuff,[byte(rs)...)
897   inEsc = 0
898   continue;
899   default:
900     // postpone the interpretation
901     //rstr = rstr + "%" + string(ch)
902   rbuff = append(rbuff,ch)
903   inEsc = 0
904   continue;
905   }
906   inEsc = 0
907 }
908 //rstr = rstr + string(ch)
909 rbuff = append(rbuff,ch)
910 }
911 //fmt.Printf("--D--subst(%s)(%s)\n",str,string(rbuff))
912 return string(rbuff)
913 //return rstr
914 }
915 func showFileInfo(path string, opts []string) {
916 if isin("-l",opts) || isin("-is",opts) {
917   fi, err := os.Stat(path)
918   if err != nil {
919     fmt.Printf("----- ((%v))",err)
920   }else{
921     mod := fi.ModTime()
922     date := mod.Format(time.Stamp)
923     fmt.Printf("%v %v %s ",fi.Mode(),fi.Size(),date)
924   }
925   fmt.Printf("%s",path)
926 if isin("-sp",opts) {
927   fmt.Println(" ")
928 }else
929 if ! isin("-n",opts) {
930   fmt.Println("\n")
931 }
932 }
933 }
934 func userHomeDir()(string,bool){
935 /*
936 homedir,_ = os.UserHomeDir() // not implemented in older Golang
937 */
938 homedir,found := os.LookupEnv("HOME")
939 //fmt.Println("--I-- HOME=%v(%v)\n",homedir,found)
940 if !found {
941   return "/tmp",found
942 }
943 return homedir,found
944 }
945
946 func toFullpath(path string) (fullpath string) {
947 if path[0] == '/' {
948   return path
949 }
950 pathv := strings.Split(path,DIRSEP)
951 switch {
952 case pathv[0] == ".":
953   pathv[0], _ = os.Getwd()
954 case pathv[0] == "..": // all ones should be interpreted
955   cwd, _ := os.Getwd()
956   ppavh := strings.Split(cwd,DIRSEP)
957   pathv[0] = strings.Join(ppavh,DIRSEP)
958 case pathv[0] == "~":
959   pathv[0],_ = userHomeDir()
960 default:
961   cwd, _ := os.Getwd()
962   pathv[0] = cwd + DIRSEP + pathv[0]
963 }
964 return strings.Join(pathv,DIRSEP)
965 }
966
967 func IsRegFile(path string)(bool){
968   fi, err := os.Stat(path)
969   if err == nil {
970     fm := fi.Mode()
971     return fm.IsRegular();
972   }
973   return false
974 }
975
976 // <a name="encode">Encode / Decode</a>
977 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
978 func (gshCtx *GshContext)Enc(argv[]string){
979   file := os.Stdin
980   buff := make([]byte,LINESIZE)
981   li := 0
982   encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
983   for li = 0; ; li++ {
984     count, err := file.Read(buff)
985     if count <= 0 {
986       break
987     }
988     if err != nil {
989       break
990     }
991     encoder.Write(buff[0:count])
992   }
}

```

```

993     encoder.Close()
994 }
995 func (gshCtx *GshContext)Dec(argv[]string){
996     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
997     li := 0
998     buff := make([]byte,LINESIZE)
999     for li = 0; ; li++ {
1000         count, err := decoder.Read(buff)
1001         if count <= 0 {
1002             break
1003         }
1004         if err != nil {
1005             break
1006         }
1007         os.Stdout.Write(buff[0:count])
1008     }
1009 }
1010 // lns [N] [-crlf]{-C \\}
1011 func (gshCtx *GshContext)SplitLine(argv[]string){
1012     strRep := isin("-str",argv) // "..."+
1013     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1014     ni := 0
1015    toi := 0
1016     for ni = 0; ; ni++ {
1017         line, err := reader.ReadString('\n')
1018         if len(line) <= 0 {
1019             if err != nil {
1020                 fmt.Fprintf(os.Stderr,"--I-- lns %d to %d (%v)\n",ni,toi,err)
1021                 break
1022             }
1023         }
1024         off := 0
1025         ilen := len(line)
1026         remlen := len(line)
1027         if strRep { os.Stdout.Write([]byte("\n")) }
1028         for oi := 0; 0 < remlen; oi++ {
1029             olen := remlen
1030             addnl := false
1031             if 72 < olen {
1032                 olen = 72
1033                 addnl = true
1034             }
1035             fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d%d\n",
1036                         toi,ni,oi,off,olen,remlen,ilen)
1037             toi += 1
1038             os.Stdout.Write([]byte(line[0:olen]))
1039             if addnl {
1040                 if strRep {
1041                     os.Stdout.Write([]byte("\r\n"))
1042                 }else{
1043                     //os.Stdout.Write([]byte("\r\n"))
1044                     os.Stdout.Write([]byte("\\"))
1045                     os.Stdout.Write([]byte("\n"))
1046                 }
1047             }
1048             line = line[olen:]
1049             off += olen
1050             remlen -= olen
1051         }
1052         if strRep { os.Stdout.Write([]byte("\r\n")) }
1053     }
1054     fmt.Fprintf(os.Stderr,"--I-- lns %d to %d\n",ni,toi)
1055 }
1056
1057 // CRC32 <a href="http://golang.jp/pkg/hash-crc32">crc32</a>
1058 // 1 0000 0100 1100 0001 0001 1101 1011 0111
1059 var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
1060 var CRC32IEEE uint32 = uint32(0xEDB88320)
1061 func byteCRC32add(crc uint32,str[]byte,len uint64)(uint32){
1062     var oi uint64
1063     for oi = 0; oi < len; oi++ {
1064         var oct = str[oi]
1065         for bi := 0; bi < 8; bi++ {
1066             //fprintf(stderr,"--CRC32 %d %X (%d.%d)\n",crc,oct,oi,bi)
1067             ovf1 := (crc & 0x80000000) != 0
1068             ovf2 := (oct & 0x80) != 0
1069             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
1070             oct <<= 1
1071             crc <<= 1
1072             if ovf { crc ^= CRC32UNIX }
1073         }
1074     }
1075     //fprintf(stderr,"--CRC32 return %d %d\n",crc,len)
1076     return crc;
1077 }
1078 func byteCRC32end(crc uint32, len uint64)(uint32){
1079     var slen = make([]byte,4)
1080     var li = 0
1081     for li = 0; li < 4; {
1082         slen[li] = byte(len)
1083         li += 1
1084         len >= 8
1085         if( len == 0 ){
1086             break
1087         }
1088     }
1089     crc = byteCRC32add(crc,slen,uint64(li))
1090     crc ^= 0xFFFFFFFF
1091     return crc
1092 }
1093 func strCRC32(str string,len uint64)(crc uint32){
1094     crc = byteCRC32add(0,[]byte(str),len)
1095     crc = byteCRC32end(crc,len)
1096     //fprintf(stderr,"--CRC32 %d %d\n",crc,len)
1097     return crc
1098 }
1099 func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
1100     var slen = make([]byte,4)
1101     var li = 0
1102     for li = 0; li < 4; {
1103         slen[li] = byte(len & 0xFF)
1104         li += 1
1105         len >= 8
1106         if( len == 0 ){
1107             break
1108         }
1109     }
1110     crc = crc32.Update(crc,table,slen)
1111     crc ^= 0xFFFFFFFF
1112     return crc
1113 }
1114
1115 func (gsh*GshContext)xCksum(path string,argv[]string, sum*CheckSum)(int64){
1116     if isin("-type/f",argv) && !IsRegFile(path){

```

```

1117     return 0
1118 }
1119 if isin("-type/d", argv) && IsRegFile(path){
1120     return 0
1121 }
1122 file, err := os.OpenFile(path,os.O_RDONLY,0)
1123 if err != nil {
1124     fmt.Printf("--E-- cksum %v (%v)\n",path,err)
1125     return -1
1126 }
1127 defer file.Close()
1128 if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n",path,argv) }
1129
1130 bi := 0
1131 var buff = make([]byte,32*1024)
1132 var total int64 = 0
1133 var initTime = time.Time()
1134 if sum.Start == initTime {
1135     sum.Start = time.Now()
1136 }
1137 for bi = 0; ; bi++ {
1138     count,err := file.Read(buff)
1139     if count <= 0 || err != nil {
1140         break
1141     }
1142     if (sum.SumType & SUM_SUM64) != 0 {
1143         s := sum.Sum64
1144         for _,c := range buff[0:count] {
1145             s += uint64(c)
1146         }
1147         sum.Sum64 = s
1148     }
1149     if (sum.SumType & SUM_UNIXFILE) != 0 {
1150         sum.Crc32Val = byteCRC32add(sum.Crc32Val,buff,uint64(count))
1151     }
1152     if (sum.SumType & SUM_CRCIEEE) != 0 {
1153         sum.Crc32Val = crc32.Update(sum.Crc32Val,&sum.Crc32Table,buff[0:count])
1154     }
1155 // <a href="https://en.wikipedia.org/wiki/BSB_checksum">BSD checksum</a>
1156 if (sum.SumType & SUM_SUM16_BSD) != 0 {
1157     s := sum.Sum16
1158     for _,c := range buff[0:count] {
1159         s = (s >> 1) + ((s & 1) << 15)
1160         s += int(c)
1161         s &= 0xFFFF
1162         //fmt.Printf("BSDsum: %d %d\n",sum.Size+int64(i),i,s)
1163     }
1164     sum.Sum16 = s
1165 }
1166 if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1167     for bj := 0; bj < count; bj++ {
1168         sum.Sum16 += int(buff[bj])
1169     }
1170 }
1171 total += int64(count)
1172 }
1173 sum.Done = time.Now()
1174 sum.Files += 1
1175 sum.Size += total
1176 if !isin("-s",argv) {
1177     fmt.Printf("%v ",total)
1178 }
1179
1180 }
1181
1182 // <a name="grep">grep</a>
1183 // "lines", "lin" or "inp" for "(text) line processor" or "scanner"
1184 // a*,!ab,c, ... sequenitioal combination of patterns
1185 // what "LINE" is should be definable
1186 // generic line-by-line processing
1187 // grep [-v]
1188 // cat -n -v
1189 // uniq [-c]
1190 // tail -f
1191 // sed s/x/y/ or awk
1192 // grep with line count like wc
1193 // rewrite contents if specified
1194 func (gsh*GshContext)xGrep(path string,rexpv[]string)(int){
1195     file, err := os.OpenFile(path,os.O_RDONLY,0)
1196     if err != nil {
1197         fmt.Printf("--E-- grep %v (%v)\n",path,err)
1198         return -1
1199     }
1200     defer file.Close()
1201     if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n",path,rexpv) }
1202     //reader := bufio.NewReaderSize(file,LINESIZE)
1203     reader := bufio.NewReaderSize(file,80)
1204     li := 0
1205     found := 0
1206     for li = 0; ; li++ {
1207         line, err := reader.ReadString('\n')
1208         if len(line) <= 0 {
1209             break
1210         }
1211         if 150 < len(line) {
1212             // maybe binary
1213             break;
1214         }
1215         if err != nil {
1216             break
1217         }
1218         if 0 <= strings.Index(string(line),rexpv[0]) {
1219             found += 1
1220             fmt.Printf("%s:%d: %s",path,li,line)
1221         }
1222     }
1223     //fmt.Printf("total %d lines %s\n",li,path)
1224 //if( 0 < found){ fmt.Printf("(found %d lines %s)\n",found,path); }
1225 return found
1226 }
1227
1228 // <a name="finder">Finder</a>
1229 // finding files with it name and contents
1230 // file names are ORed
1231 // show the content with %x fmt list
1232 // ls -R
1233 // tar command by adding output
1234 type fileSum struct {
1235     Err int64 // access error or so
1236     Size int64 // content size
1237     DupSize int64 // content size from hard links
1238     Blocks int64 // number of blocks (of 512 bytes)
1239     DupBlocks int64 // Blocks pointed from hard links
1240     HLinks int64 // hard links

```

```

1241     Words   int64
1242     Lines   int64
1243     Files   int64
1244     Dirs    int64 // the num. of directories
1245     SymLink int64
1246     Flats   int64 // the num. of flat files
1247     MaxDepth int64
1248     MaxNameLen int64 // max. name length
1249     nextRepo time.Time
1250 }
1251 func showUsage(dir string,fusage *fileSum){
1252     bsum := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1253     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1254
1255     fmt.Printf("%v: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
1256             dir,
1257             fusage.Files,
1258             fusage.Dirs,
1259             fusage.SymLink,
1260             fusage.HLinks,
1261             float64(fusage.Size)/1000000.0,bsum);
1262 }
1263 const (
1264     S_IFMT    = 0170000
1265     S_IFCHR   = 0020000
1266     S_IFDIR   = 0040000
1267     S_IFREG   = 0100000
1268     S_IFLNK   = 0120000
1269     S_IFSOCK  = 0140000
1270 )
1271 func cumFinfo(fsum *fileSum, path string, staterr error, fstat syscall.Stat_t, argv[]string, verb bool)(*fileSum{
1272     now := time.Now()
1273     if time.Second <= now.Sub(fsum.nextRepo) {
1274         if !fsum.nextRepo.IsZero(){
1275             tstamp := now.Format(time.Stamp)
1276             showUsage(tstamp,fsum)
1277         }
1278         fsum.nextRepo = now.Add(time.Second)
1279     }
1280     if staterr != nil {
1281         fsum.Err += 1
1282         return fsum
1283     }
1284     fsum.Files += 1
1285     if 1 < fstat.Nlink {
1286         // must count only once...
1287         // at least ignore ones in the same directory
1288         //if finfo.Mode().IsRegular() {
1289         if (fstat.Mode & S_IFMT) == S_IFREG {
1290             fsum.HLinks += 1
1291             fsum.DupBlocks += int64(fstat.Blocks)
1292             //fmt.Printf("---Dup HardLink %v %s\n",fstat.Nlink,path)
1293         }
1294     }
1295     //fsum.Size += finfo.Size()
1296     fsum.Size += fstat.Size()
1297     fsum.Blocks += int64(fstat.Blocks)
1298     //if verb { fmt.Printf("(8dBlk) %s",fstat.Blocks/2,path) }
1299     if isin("-ls",argv){
1300         //if verb { fmt.Printf("%4d %8d ",fstat.Blksize,fstat.Blocks) }
1301     }
1302     //if finfo.IsDir()
1303     if (fstat.Mode & S_IFMT) == S_IFDIR {
1304         fsum.Dirs += 1
1305     }
1306     //if (finfo.Mode() & os.ModeSymlink) != 0
1307     if (fstat.Mode & S_IFMT) == S_IFLINK {
1308         //if verb { fmt.Printf("symlink(%v,%s)\n",fstat.Mode,finfo.Name()) }
1309         //fmt.Printf("symlink(%o,%s)\n",fstat.Mode,finfo.Name())
1310         fsum.Symlink += 1
1311     }
1312     return fsum
1313 }
1314 }
1315 func (gsh*GshContext)xxFindEnv(depth int,total *fileSum,dir string, dstat syscall.Stat_t, ei int, envt []string,npatv[]string,argv[]string)(*fileSum{
1316     nols := isin("-grep",argv)
1317     // sort envt
1318     /*
1319     if isin("-t",argv){
1320         sort.Slice(filev, func(i,j int) bool {
1321             return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
1322         })
1323     }
1324     */
1325     /*
1326     if isin("-u",argv){
1327         sort.Slice(filev, func(i,j int) bool {
1328             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1329         })
1330     }
1331     if isin("-U",argv){
1332         sort.Slice(filev, func(i,j int) bool {
1333             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1334         })
1335     }
1336     */
1337     /*
1338     if isin("-S",argv){
1339         sort.Slice(filev, func(i,j int) bool {
1340             return filev[j].Size() < filev[i].Size()
1341         })
1342     }
1343     */
1344     for _,filename := range envt {
1345         for _,npat := range npatv {
1346             match := true
1347             if npat == "*" {
1348                 match = true
1349             }else{
1350                 match, _ = filepath.Match(npatt,filename)
1351             }
1352             path := dir + DIRSEP + filename
1353             if !match {
1354                 continue
1355             }
1356             var fstat syscall.Stat_t
1357             staterr := syscall.Lstat(path,&fstat)
1358             if staterr != nil {
1359                 if !isin("-w",argv){fmt.Printf("ufind: %v\n",staterr) }
1360                 continue;
1361             }
1362             if isin("-du",argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
1363                 // should not show size of directory in "-du" mode ...
1364             }else

```

```

1365     if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1366         if isin("-du",argv) {
1367             fmt.Printf("%d\t",fstat.Blocks/2)
1368         }
1369         showFileInfo(path,argv)
1370     }
1371     if true { // && isin("-du",argv)
1372         total = cumFileInfo(total,path,staterr,fstat,argv,false)
1373     }
1374     /*
1375     if isin("-wc",argv) {
1376     */
1377     if gsh.lastCheckSum.SumType != 0 {
1378         gsh.xCksum(path,argv,&gsh.lastCheckSum);
1379     }
1380     x := isinX("-grep",argv); // -grep will be convenient like -ls
1381     if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
1382         if IRegFile(path){
1383             found := gsh.xGrep(path,argv[x+1])
1384             if 0 < found {
1385                 foundv := gsh.CmdCurrent.FoundFile
1386                 if len(foundv) < 10 {
1387                     gsh.CmdCurrent.FoundFile =
1388                         append(gsh.CmdCurrent.FoundFile,path)
1389                 }
1390             }
1391         }
1392     }
1393     if !isin("-r0",argv) { // -d 0 in du, -depth n in find
1394         //total.Depth += 1
1395         if (fstat.Mode & S_IFMT) == S_IFLNK {
1396             continue
1397         }
1398         if dstat.Rdev != fstat.Rdev {
1399             fmt.Printf("--I-- don't follow differnet device %v(%v) %v(%v)\n",
1400                     dir,dstat.Rdev,path,fstat.Rdev)
1401         }
1402         if (fstat.Mode & S_IFMT) == S_IFDIR {
1403             total = gsh.xxFind(depth+1,total,path,npatv,argv)
1404         }
1405     }
1406 }
1407 }
1408 }
1409 return total
1410 }
1411 func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npatv[]string,argv[]string)(*fileSum{
1412 nols := isin("-grep",argv)
1413 dirfile,err := os.OpenFile(dir,os.O_RDONLY,0)
1414 if err == nil {
1415     //fmt.Printf("--I-- %v(%v)[%d]\n",dir,dirfile,dirfile.Fd())
1416     defer dirfile.Close()
1417 }else{
1418 }
1419 prev := *total
1420 var dstat syscall.Stat_t
1421 staterr := syscall.Lstat(dir,&dstat) // should be fstat
1422
1423 if staterr != nil {
1424     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",staterr) }
1425     return total
1426 }
1427 //filev,err := ioutil.ReadDir(dir)
1428 //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1429 /*
1430 if err != nil {
1431     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1432     return total
1433 }
1434 */
1435 if depth == 0 {
1436     total = cumFileInfo(total,dir,staterr,dstat,argv,true)
1437     if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1438         showFileInfo(dir,argv)
1439     }
1440 }
1441 // it it is not a directory, just scan it and finish
1442 for ei := 0; ei++ {
1443     entv,rdrerr := dirfile.Readdirnames(8*1024)
1444     if len(entv) == 0 || rdrerr != nil {
1445         //if rdrerr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),rdrerr) }
1446         break
1447     }
1448     if 0 < ei {
1449         fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
1450     }
1451     total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
1452 }
1453 if isin("-du",argv) {
1454     // if in "du" mode
1455     fmt.Printf("%d\t%s\n", (total.Blocks-prev.Blocks)/2,dir)
1456 }
1457 return total
1458 }
1459 }
1460 }
1461
1462 // {ufind|ls} [Files] [-- Expressions]
1463 // Files is "-" by default
1464 // Names is "*" by default
1465 // Expressions is "-print" by default for "ufind", or -du for "fu" command
1466 func (gsh*GshContext)xFind(argv[]string){
1467 if 0 < len(argv) && strBegins(argv[0],"?"){
1468     showFound(gsh,argv)
1469     return
1470 }
1471 if isin("-cksum",argv) || isin("-sum",argv) {
1472     gsh.lastCheckSum = CheckSum{}
1473     if isin("-sum",argv) && isin("-add",argv) {
1474         gsh.lastCheckSum.SumType |= SUM_SUM64
1475     }
1476     if isin("-sum",argv) && isin("-size",argv) {
1477         gsh.lastCheckSum.SumType |= SUM_SIZE
1478     }
1479     if isin("-sum",argv) && isin("-bsd",argv) {
1480         gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
1481     }
1482     if isin("-sum",argv) && isin("-sysv",argv) {
1483         gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
1484     }
1485     if isin("-sum",argv) {
1486         gsh.lastCheckSum.SumType |= SUM_SUM64
1487     }
1488     if isin("-unix",argv) {
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
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
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347

```

```

1489         gsh.lastCheckSum.SumType |= SUM_UNIXFILE
1490         gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32UNIX)
1491     } if isin("-ieee", argv){
1492         gsh.lastCheckSum.SumType |= SUM_CRCIEEE
1493         gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32IEEE)
1494     } gsh.lastCheckSum.RusgAtStart = Getrusagev()
1495 }
1496 var total = fileSum()
1497 npats := []string{}
1498 for _, v := range argv {
1499     if 0 < len(v) && v[0] != '-' {
1500         npats = append(npats, v)
1501     }
1502     if v == "//" { break }
1503     if v == "-" { break }
1504     if v == "-grep" { break }
1505     if v == "-ls" { break }
1506 }
1507 if len(npats) == 0 {
1508     npats = []string{"*"}
1509 }
1510 cwd := "."
1511 // if to be fullpath :: cwd, _ := os.Getwd()
1512 if len(npats) == 0 { npats = []string{*} }
1513 fusage := gsh.xxFind(0, &total, cwd, npats, argv)
1514 if gsh.lastCheckSum.SumType != 0 {
1515     var sumi uint64 = 0
1516     sum := &gsh.lastCheckSum
1517     if (sum.SumType & SUM_SIZE) != 0 {
1518         sumi = uint64(sum.Size)
1519     }
1520     if (sum.SumType & SUM_SUM64) != 0 {
1521         sumi = sum.Sum64
1522     }
1523     if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1524         s := uint32(sum.Sum16)
1525         r := (s & 0xFFFF) + ((s & 0xFFFFFFF) >> 16)
1526         s = (r & 0xFFFF) + (r >> 16)
1527         sum.Crc32Val = uint32(s)
1528         sumi = uint64(s)
1529     }
1530     if (sum.SumType & SUM_SUM16_BSD) != 0 {
1531         sum.Crc32Val = uint32(sum.Sum16)
1532         sumi = uint64(sum.Sum16)
1533     }
1534     if (sum.SumType & SUM_UNIXFILE) != 0 {
1535         sum.Crc32Val = byteCRC32end(sum.Crc32Val, uint64(sum.Size))
1536         sumi = uint64(byteCRC32end(sum.Crc32Val, uint64(sum.Size)))
1537     }
1538     if 1 < sum.Files {
1539         fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
1540                     sumi, sum.Size,
1541                     abssize(sum.Size), sum.Files,
1542                     abssize(sum.Size/sum.Files))
1543     } else{
1544         fmt.Printf("%v %v %v\r\n",
1545                     sumi, sum.Size, npats[0])
1546     }
1547 }
1548 if !isin("-grep", argv) {
1549     showUsage("total", fusage)
1550 }
1551 if !isin("-s", argv){
1552     hits := len(gsh.CmdCurrent.FoundFile)
1553     if 0 < hits {
1554         fmt.Printf("--I-- %d files hits // can be refered with !%df\n",
1555                     hits, len(gsh.CommandHistory))
1556     }
1557 }
1558 if gsh.lastCheckSum.SumType != 0 {
1559     if isin("-ru", argv) {
1560         sum := &gsh.lastCheckSum
1561         sum.Done = time.Now()
1562         gsh.lastCheckSum.RusgAtEnd = Getrusagev()
1563         elps := sum.Done.Sub(sum.Start)
1564         fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\r\n",
1565                     sum.Size, abssize(sum.Size), sum.Files, abssize(sum.Size/sum.Files))
1566         nanos := int64(elps)
1567         fmt.Printf("--cksum-time: %v/total, %v/file, %.lf files/s, %v/r\n",
1568                     abbtme(nanos),
1569                     abbtme(nanos/sum.Files),
1570                     float64(sum.Files)*1000000000.0/float64(nanos),
1571                     abbspeed(sum.Size,nanos))
1572         diff := RusageSubv(sum.RusgAtEnd,sum.RusgAtStart)
1573         fmt.Println("--cksum-rusg: %v\n",skusagef(argv,diff))
1574     }
1575 }
1576 }
1577 return
1578 }
1579 }
1580
1581 func showFiles(files[]string{
1582     sp := ""
1583     for i,file := range files {
1584         if 0 < i { sp = " " } else { sp = "" }
1585         fmt.Printf(sp+"%s",escapeWhiteSP(file))
1586     }
1587 }
1588 func showFound(gshCtx *GshContext, argv[]string{
1589     for i,v := range gshCtx.CommandHistory {
1590         if 0 < len(v.Foundfile) {
1591             fmt.Printf("%#d ",i,len(v.Foundfile))
1592             if isin("-ls", argv){
1593                 fmt.Printf("\n")
1594                 for _,file := range v.Foundfile {
1595                     fmt.Printf("%") //sub number?
1596                     showFileInfo(file,argv)
1597                 }
1598             } else{
1599                 showFiles(v.Foundfile)
1600                 fmt.Printf("\n")
1601             }
1602         }
1603     }
1604 }
1605
1606 func showMatchFile(filev []os.FileInfo, npat,dir string, argv[]string)(string,bool){
1607     fname := ""
1608     found := false
1609     for _,v := range filev {
1610         match, _ := filepath.Match(npata,(v.Name()))
1611         if match {
1612             fname = v.Name()
1613         }
1614     }
1615 }

```

```

1613     found = true
1614     //fmt.Printf("[%d] %s\n",i,v.Name())
1615     showIfExecutable(fname,dir,argv)
1616   }
1617 }
1618 return fname,found
1619 }
1620 func showIfExecutable(name,dir string,argv[]string)(ffullpath string,ffound bool){
1621 var fullpath string
1622 if strBegins(name,DIRSEP){
1623   fullpath = name
1624 }else{
1625   fullpath = dir + DIRSEP + name
1626 }
1627 fi, err := os.Stat(fullpath)
1628 if err != nil {
1629   fullpath = dir + DIRSEP + name + ".go"
1630   fi, err = os.Stat(fullpath)
1631 }
1632 if err == nil {
1633   fm := fi.Mode()
1634   if fm.IsRegular() {
1635     // R_OK=4, W_OK=2, X_OK=1, F_OK=0
1636     if syscall.Access(fullpath,5) == nil {
1637       ffullpath = fullpath
1638       ffound = true
1639       if ! isin("-s", argv) {
1640         showFileInfo(ffullpath,argv)
1641       }
1642     }
1643   }
1644 }
1645 return ffullpath,ffound
1646 }
1647 func which(list string, argv []string) (fullpathv []string, itis bool){
1648 if len(argv) <= 1 {
1649   fmt.Printf("Usage: which [command [-s] [-a] [-ls]]\n")
1650   return []string{}, false
1651 }
1652 path := argv[1]
1653 if strBegins(path,"/") {
1654   // should check if executable?
1655   _exOK := showIfExecutable(path,"/",argv)
1656   fmt.Printf("--D--- %v exOK=%v\n",path,_exOK)
1657   return []string{path},_exOK
1658 }
1659 pathenv, efound := os.LookupEnv(list)
1660 if ! efound {
1661   fmt.Printf("--E-- which: no \"%s\" environment\n",list)
1662   return []string{}, false
1663 }
1664 showall := isin("-a",argv) || 0 <= strings.Index(path,"*")
1665 dirv := strings.Split(pathenv,PATHSEP)
1666 ffound := false
1667 ffullpath := path
1668 for _, dir := range dirv {
1669   if 0 <= strings.Index(path,"*") { // by wild-card
1670     list,_ := ioutil.ReadDir(dir)
1671     ffullpath,ffound = showMatchFile(list,path,dir,argv)
1672   }else{
1673     ffullpath,ffound = showIfExecutable(path,dir,argv)
1674   }
1675 //if ffound && !isin("-a", argv) {
1676 if ffound && !showall {
1677   if !showall {
1678     break;
1679   }
1680 }
1681 return []string{ffullpath},ffound
1682 }
1683 func stripLeadingWSParg(argv[]string)([]string){
1684 for ; 0 < len(argv); {
1685   if len(argv[0]) == 0 {
1686     argv = argv[1:]
1687   }else{
1688     break
1689   }
1690 }
1691 return argv
1692 }
1693 func xEval(argv []string, nlend bool){
1694 argv = stripLeadingWSParg(argv)
1695 if len(argv) == 0 {
1696   fmt.Printf("eval [%s] [Go-expression]\n"
1697   return
1698 }
1699 pfmt := "%v"
1700 if argv[0][0] == '%' {
1701   pfmt = argv[0]
1702   argv = argv[1:]
1703 }
1704 if len(argv) == 0 {
1705   return
1706 }
1707 gocode := strings.Join(argv, " ");
1708 //fmt.Printf("eval [%v] [%v]\n",pfmt,gocode)
1709 fset := token.NewfileSet()
1710 rval, _ := types.Eval(fset,nil,token.NoPos,gocode)
1711 fmt.Printf(pfmt,rval.Value)
1712 if nlend { fmt.Println("\n") }
1713 }
1714 }
1715 func getval(name string) (found bool, val int) {
1716 /* should expand the name here */
1717 if name == "gsh.pid" {
1718   return true, os.Getpid()
1719 }else{
1720 if name == "gsh.ppid" {
1721   return true, os.Getppid()
1722 }
1723 return false, 0
1724 }
1725 }
1726 func echo(argv []string, nlend bool){
1727 for ai := 1; ai < len(argv); ai++ {
1728   if 1 < ai {
1729     fmt.Printf(" ")
1730   }
1731   arg := argv[ai]
1732   found, val := getval(arg)
1733   if found {
1734     fmt.Printf("%d",val)
1735   }else{
1736     fmt.Printf("%s",arg)
1737   }
1738 }
1739 }

```

```

1737     }
1738 }
1739 if nlen {
1740     fmt.Printf("\n");
1741 }
1742 }
1743
1744 func resfile() string {
1745     return "gsh.tmp"
1746 }
1747 //var resF *File
1748 func resmap() {
1749     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
1750     // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
1751     // , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
1752     if err != nil {
1753         fmt.Printf("resF could not open: %s\n",err)
1754     }else{
1755         fmt.Printf("resF opened\n")
1756     }
1757 }
1758
1759 // @@2020-0821
1760 func gshScanArg(str string,strip int)(argv []string){
1761     var si = 0
1762     var sb = 0
1763     var inBracket = 0
1764     var argl = make([]byte,LINESIZE)
1765     var ax = 0
1766     debug := false
1767
1768     for ; si < len(str); si++ {
1769         if str[si] != ' ' {
1770             break
1771         }
1772     }
1773     sb = si
1774     for ; si < len(str); si++ {
1775         if sb <= si {
1776             if debug {
1777                 fmt.Printf("--Da- %d %2d-%2d %s ... %s\n",
1778                     inBracket,sb,si,argl[0:ax],str[si:])
1779             }
1780         }
1781         ch := str[si]
1782         if ch == '{' {
1783             inBracket += 1
1784             if 0 < strip && inBracket <= strip {
1785                 //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
1786                 continue
1787             }
1788         if 0 < inBracket {
1789             if ch == ')' {
1790                 inBracket -= 1
1791                 if 0 < strip && inBracket < strip {
1792                     //fmt.Printf("stripLEV %d < %d?\n",inBracket,strip)
1793                     continue
1794                 }
1795             }
1796             argl[ax] = ch
1797             ax += 1
1798             continue
1799         }
1800     }
1801     if str[si] == ' ' {
1802         argv = append(argv,string(argl[0:ax]))
1803         if debug {
1804             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
1805                 -1+len(argv),sb,si,str[sb:si],string(str[si:]))
1806         }
1807         sb = si+1
1808         ax = 0
1809         continue
1810     }
1811     argl[ax] = ch
1812     ax += 1
1813 }
1814 if sb < si {
1815     argv = append(argv,string(argl[0:ax]))
1816     if debug {
1817         fmt.Printf("--Da- [%v]{%v-%v} %s ... %s\n",
1818             -1+len(argv),sb,si,string(argl[0:ax]),string(str[si:]))
1819     }
1820 }
1821 if debug {
1822     fmt.Printf("--Da- %d [%s] => [%d]%v\n",strip,str,len(argv),argv)
1823 }
1824 return argv
1825 }
1826
1827 // should get stderr (into tmpfile ?) and return
1828 func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
1829     var pv = []int{-1,-1}
1830     syscall.Pipe(pv)
1831
1832     xarg := gshScanArg(name,1)
1833     name = strings.Join(xarg, " ")
1834
1835     pin = os.NewFile(uintptr(pv[0]),"StdoutOf-{"+name+"}")
1836     pout = os.NewFile(uintptr(pv[1]),"StdinOf-{"+name+"}")
1837     fidx := 0
1838     dir := "?"
1839     if mode == "r" {
1840         dir = "<"
1841         fidx = 1 // read from the stdout of the process
1842     }else{
1843         dir = ">"
1844         fidx = 0 // write to the stdin of the process
1845     }
1846     gshPA := gsh.gshPA
1847     savfd := gshPA.Files[fidx]
1848
1849     var fd uintptr = 0
1850     if mode == "r" {
1851         fd = pout.Fd()
1852         gshPA.Files[fidx] = pout.Fd()
1853     }else{
1854         fd = pin.Fd()
1855         gshPA.Files[fidx] = pin.Fd()
1856     }
1857     // should do this by Goroutine?
1858     if false {
1859         fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
1860         fmt.Printf(" --RED1 [%d,%d,%d]->[%d,%d,%d]\n",

```

```

1861         os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
1862         pin.Fd(),pout.Fd(),pout.Fd())
1863     }
1864     savi := os.Stdin
1865     savo := os.Stdout
1866     save := os.Stderr
1867     os.Stdin = pin
1868     os.Stdout = pout
1869     os.Stderr = pout
1870     gsh.BackGround = true
1871     gsh.gshellh(name)
1872     gsh.BackGround = false
1873     os.Stdin = savi
1874     os.Stdout = savo
1875     os.Stderr = save
1876
1877     gshPA.Files[fdix] = savfd
1878     return pin,pout,false
1879 }
1880
1881 // <a name="ex-commands">External commands</a>
1882 func (gsh*GshContext)execCommand(exec bool, argv []string) (notf bool,exit bool) {
1883     if gsh.CmdTrace { fmt.Printf("--I-- execcommand[%v](%v)\n",exec,argv) }
1884
1885     gshPA := gsh.gshPA
1886     fullpathv, itis := which("PATH",[]string{"which",argv[0],"-s"})
1887     if itis == false {
1888         return true,false
1889     }
1890     fullpath := fullpathv[0]
1891     argv = unescapeWhiteSPV(argv)
1892     if 0 < strings.Index(fullpath,".") {
1893         nargv := argv // ([]string{
1894         gofullpathv, itis := which("PATH",[]string{"which",".","-s"})
1895         if itis == false {
1896             fmt.Printf("--F-- Go not found\n")
1897             return false,true
1898         }
1899         gofullpath := gofullpathv[0]
1900         nargv = []string{gofullpath, "run", fullpath }
1901         fmt.Printf("--I-- %s %s\n",gofullpath,
1902             nargv[0],nargv[1],nargv[2])
1903         if exec {
1904             syscall.Exec(gofullpath,nargv,os.Environ())
1905         }else{
1906             pid, _ := syscall.ForkExec(gofullpath,nargv,&gshPA)
1907             if gsh.BackGround {
1908                 fmt.Fprintf(stderr,"--Ip- in Background pid[%d]%d(%v)\n",pid,len(argv),nargv)
1909                 gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
1910             }else{
1911                 rusage := syscall.Rusage {}
1912                 syscall.Wait4(pid,nil,0,&rusage)
1913                 gsh.LastRusage = rusage
1914                 gsh.CmdCurrent.Rusagev[1] = rusage
1915             }
1916         }else{
1917             if exec {
1918                 syscall.Exec(fullpath,argv,os.Environ())
1919             }else{
1920                 pid, _ := syscall.ForkExec(fullpath,argv,&gshPA)
1921                 //fmt.Printf("%d\n",pid); // '&' to be background
1922                 if gsh.BackGround {
1923                     fmt.Fprintf(stderr,"--Ip- in Background pid[%d]%d(%v)\n",pid,len(argv),argv)
1924                     gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
1925                 }else{
1926                     rusage := syscall.Rusage {}
1927                     syscall.Wait4(pid,nil,0,&rusage);
1928                     gsh.LastRusage = rusage
1929                     gsh.CmdCurrent.Rusagev[1] = rusage
1930                 }
1931             }
1932         }
1933     }
1934     return false,false
1935 }
1936
1937 // <a name="builtin">Builtin Commands</a>
1938 func (gshCtx *GshContext) sleep(argv []string) {
1939     if len(argv) < 2 {
1940         fmt.Printf("Sleep 100ms, 100us, 100ns, ...\n")
1941         return
1942     }
1943     duration := argv[1];
1944     d, err := time.ParseDuration(duration)
1945     if err != nil {
1946         d, err = time.ParseDuration(duration+"s")
1947         if err != nil {
1948             fmt.Printf("duration ? %s (%s)\n",duration,err)
1949             return
1950         }
1951     }
1952     //fmt.Printf("Sleep %v\n",duration)
1953     time.Sleep(d)
1954     if 0 < len(argv[2:]) {
1955         gshCtx.gshellv(argv[2:])
1956     }
1957 }
1958 func (gshCtx *GshContext)repeat(argv []string) {
1959     if len(argv) < 2 {
1960         return
1961     }
1962     start0 := time.Now()
1963     for ri,_ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
1964         if 0 < len(argv[2:]) {
1965             //start := time.Now()
1966             gshCtx.gshellv(argv[2:])
1967             end := time.Now()
1968             elps := end.Sub(start0);
1969             if( 1000000000 < elps ){
1970                 fmt.Printf("(repeat#%d %v)\n",ri,elps);
1971             }
1972         }
1973     }
1974 }
1975
1976 func (gshCtx *GshContext)gen(argv []string) {
1977     gshPA := gshCtx.gshPA
1978     if len(argv) < 2 {
1979         fmt.Printf("Usage: %s N\n",argv[0])
1980         return
1981     }
1982     // should br repeated by "repeat" command
1983     count, _ := strconv.Atoi(argv[1])
1984     fd := gshPA.Files[1] // Stdout

```

```

1985     file := os.NewFile(fd,"internalStdOut")
1986     fmt.Printf("--I-- Gen. Count=%d to [%d]\n",count,file.Fd())
1987     //buf := []byte{}
1988     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
1989     for gi := 0; gi < count; gi++ {
1990         file.WriteString(outdata)
1991     }
1992     //file.WriteString("\n")
1993     fmt.Printf("\n(%d B)\n",count*len(outdata));
1994     //file.Close()
1995 }
1996
1997 // <a name="rexec">Remote Execution</a> // 2020-0820
1998 func Elapsed(from time.Time)(string){
1999     elps := time.Now().Sub(from)
2000     if 1000000000 < elps {
2001         return fmt.Sprintf("[%5d.%02ds]",elps/1000000000,(elps%100000000)/1000000)
2002     }else{
2003         if 1000000 < elps {
2004             return fmt.Sprintf("%3d.%03dms",elps/1000000,(elps%1000000)/1000)
2005         }else{
2006             return fmt.Sprintf("%3d.%03dus",elps/1000,(elps%1000))
2007         }
2008 }
2009 func abftime(nanos int64)(string){
2010     if 1000000000 < nanos {
2011         return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%1000000000)/1000000)
2012     }else{
2013         if 1000000 < nanos {
2014             return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
2015         }else{
2016             return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
2017         }
2018 }
2019 func abssize(size int64)(string){
2020     fsize := float64(size)
2021     if 1024*1024*1024 < size {
2022         return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
2023     }else{
2024         if 1024*1024 < size {
2025             return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
2026         }else{
2027             return fmt.Sprintf("%.3fKiB",fsize/1024)
2028         }
2029 }
2030 func abszise(size int64)(string){
2031     fsize := float64(size)
2032     if 1024*1024*1024 < size {
2033         return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
2034     }else{
2035         if 1024*1024 < size {
2036             return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
2037         }else{
2038             return fmt.Sprintf("%.3fKiB",fsize/1024)
2039         }
2040 }
2041 func abbspeed(totalB int64,ns int64)(string){
2042     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2043     if 1000 <= MBs {
2044         return fmt.Sprintf("%6.3fGB/s",MBs/1000)
2045     }
2046     if 1 <= MBs {
2047         return fmt.Sprintf("%6.3fMB/s",MBs)
2048     }else{
2049         return fmt.Sprintf("%6.3fKB/s",MBs*1000)
2050     }
2051 }
2052 func absspeed(totalB int64,ns time.Duration)(string){
2053     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2054     if 1000 <= MBs {
2055         return fmt.Sprintf("%6.3fGBps",MBs/1000)
2056     }
2057     if 1 <= MBs {
2058         return fmt.Sprintf("%6.3fMBps",MBs)
2059     }else{
2060         return fmt.Sprintf("%6.3fKBps",MBs*1000)
2061     }
2062 }
2063 func fileRelay(what string,in*os.File,out*os.File,size int64,bsiz int)(wcount int64){
2064     Start := time.Now()
2065     buff := make([]byte,bsiz)
2066     var total int64 = 0
2067     var rem int64 = size
2068     nio := 0
2069     Prev := time.Now()
2070     var PrevSize int64 = 0
2071
2072     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) START\n",
2073     what,absize(total),size,nio)
2074
2075     for i:= 0; ; i++ {
2076         var len = bsiz
2077         if int(rem) < len {
2078             len = int(rem)
2079         }
2080         Now := time.Now()
2081         Elps := Now.Sub(Prev);
2082         if 1000000000 < Now.Sub(Prev) {
2083             fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %s\n",
2084             what,absize(total),size,nio,
2085             abspeed((total-PrevSize),Elps))
2086             Prev = Now;
2087             PrevSize = total
2088         }
2089         rlen := len
2090         if in != nil {
2091             // should watch the disconnection of out
2092             rcc,err := in.Read(buff[0:rlen])
2093             if err != nil {
2094                 fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<sv\n",
2095                 what,rcc,err,in.Name())
2096                 break
2097             }
2098             rlen = rcc
2099             if string(buff[0:10]) == "((SoftEOF " {
2100                 var ecc int64 = 0
2101                 fmt.Sscanf(string(buff),"(SoftEOF %v",&ecc)
2102                 fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))/%v\n",
2103                     what,ecc,total)
2104                 if ecc == total {
2105                     break
2106                 }
2107             }
2108         }

```

```

2109     wlen := rlen
2110     if out != nil {
2111         wcc,err := out.Write(buff[0:rlen])
2112         if err != nil {
2113             fmt.Printf(Elapsed(Start)+"--En- X: %s write(%v,%v)>%v\n",
2114                         what,wcc,err,out.Name())
2115             break
2116         }
2117     }
2118     wlen = wcc
2119 }
2120 if wlen < rlen {
2121     fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2122                 what,wlen,rlen)
2123     break;
2124 }
2125
2126 nio += 1
2127 total += int64(rlen)
2128 rem -= int64(rlen)
2129 if rem <= 0 {
2130     break
2131 }
2132
2133 Done := time.Now()
2134 Elps := float64(Done.Sub(Start))/1000000000 //Seconds
2135 TotalMB := float64(total)/1000000
2136 MBps := TotalMB / Elps
2137 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %v %.3fMB/s\n",
2138             what,total,size,nio,absize(total),MBps)
2139 return total
2140 }
2141 func tcpPush(clnt *os.File){
2142     // shrink socket buffer and recover
2143     usleep(100);
2144 }
2145 func (gsh*GshContext)RexecServer(argv[]string{
2146     debug : true
2147     Start0 := time.Now()
2148     Start := Start0
2149 //    if local == ":" { local = "0.0.0.0:9999" }
2150     local := "0.0.0.0:9999"
2151
2152     if 0 < len(argv) {
2153         if argv[0] == "-s" {
2154             debug = false
2155             argv = argv[1:]
2156         }
2157     }
2158     if 0 < len(argv) {
2159         argv = argv[1:]
2160     }
2161     port, err := net.ResolveTCPAddr("tcp",local);
2162     if err != nil {
2163         fmt.Printf("--En- S: Address error: %s (%s)\n",local,err)
2164         return
2165     }
2166     fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n",local);
2167     sconn, err := net.ListenTCP("tcp", port)
2168     if err != nil {
2169         fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n",local,err)
2170         return
2171     }
2172     redbuf := make([]byte,LINESIZE)
2173     res := ""
2174     for {
2175         fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
2176         aconn, err := sconn.AcceptTCP()
2177         Start = time.Now()
2178         if err != nil {
2179             fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
2180             return
2181         }
2182         clnt, _ := aconn.File()
2183         fd := clnt.Fd()
2184         ar := aconn.RemoteAddr()
2185         if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %d <- %v\n",
2186                         local,fd,ar) }
2187         res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
2188         fmt.Fprintf(clnt,"%s",res)
2189         if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
2190         count, err := clnt.Read(reqbuf)
2191         if err != nil {
2192             fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
2193                         count,err,string(reqbuf))
2194         }
2195         req := string(reqbuf[:count])
2196         if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
2197         reqv := strings.Split(string(req),"r")
2198         cmdv := gshScanArg(reqv[0],0)
2199 //cmdv := strings.Split(reqv[0]," ")
2200         switch cmdv[0] {
2201             case "HELP":
2202                 res = fmt.Sprintf("250 %v",req)
2203             case "GET":
2204                 // download (remotefile|-zN) [localfile]
2205                 var dsize int64 = 32*1024*1024
2206                 var bsize int = 64*1024
2207                 var fname string = ""
2208                 var in *os.File = nil
2209                 var pseudoEOF = false
2210                 if l < len(cmdv) {
2211                     fname = cmdv[1]
2212                     if strBegins(fname,"-z") {
2213                         fmt.Sscanf(fname[2:], "%d",&dsize)
2214                     }else{
2215                         if strBegins(fname,"(") {
2216                             xin,xout,err := gsh.Popen(fname, "r")
2217                             if err {
2218                                 }else{
2219                                     xout.Close()
2220                                     defer xin.Close()
2221                                     in = xin
2222                                     dsize = MaxStreamSize
2223                                     pseudoEOF = true
2224                                 }
2225                             }else{
2226                             xin,err := os.Open(fname)
2227                             if err != nil {
2228                                 fmt.Printf("--En- GET (%v)\n",err)
2229                             }else{
2230                                 defer xin.Close()
2231                                 in = xin
2232                             }
2233                         }
2234                     }
2235                 }
2236             }
2237         }
2238     }
2239 }
```

```

2233         fi,_ := xin.Stat()
2234         dsize = fi.Size()
2235     }
2236 }
2237 //fmt.Printf(Elapsed(Start)+"--In- GET %v:\n",dsize,bsize)
2238 res = fmt.Sprintf("200 %v\r\n",dsize)
2239 fmt.Fprintf(clnt,"%v",res)
2240 tcpPush(clnt); // should be separated as line in receiver
2241 fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2242 wcount := fileRelay("SendGET",in,clnt,dsize,bsize)
2243 if pseudoEOF {
2244     in.Close() // pipe from the command
2245     // show end of stream data (its size) by OOB?
2246     SoftEOF := fmt.Sprintf("(SoftEOF %v)",wcount)
2247     fmt.Println(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)
2248 }
2249 tcpPush(clnt); // to let SoftEOF data appear at the top of received data
2250 fmt.Fprintf(clnt,"%v\r\n",SoftEOF)
2251 tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
2252     // with client generated random?
2253     //fmt.Printf("--In- L: close %v (%v)\n",in.Fd(),in.Name())
2254 }
2255 res = fmt.Sprintf("200 GET done\r\n")
2256 case "PUT":
2257     // upload [srcfile|-zN] [dstfile]
2258     var dszie int64 = 32*1024*1024
2259     var bsize int = 64*1024
2260     var fname string = ""
2261     var out *os.File = nil
2262     if 1 < len(cmdv) { // localfile
2263         fmt.Sscanf(cmdv[1],"%d",&dszie)
2264     }
2265     if 2 < len(cmdv) {
2266         fname = cmdv[2]
2267         if fname == "-" {
2268             // nul dev
2269         }else{
2270             if strBegins(fname,"{") {
2271                 xin,xout,err := gsh.Popen(fname,"w")
2272                 if err {
2273                     xin.Close()
2274                     defer xout.Close()
2275                     out = xout
2276                 }
2277             }else{
2278                 // should write to temporary file
2279                 // should suppress ^C on tty
2280                 xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2281                 //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n",fname,xout,err)
2282                 if err != nil {
2283                     fmt.Printf("--En- PUT (%v)\n",err)
2284                 }else{
2285                     out = xout
2286                 }
2287             }
2288         }
2289         fmt.Printf(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
2290             fname,local,err)
2291     }
2292     fmt.Printf(Elapsed(Start)+"--In- PUT %v (%v)\n",dszie,bsize)
2293     fmt.Printf(Elapsed(Start)+"--In- S: 200 %v OK\r\n",dsize)
2294     fmt.Fprintf(clnt,"200 %v OK\r\n",dsize)
2295     fileRelay("RecvPUT",clnt,out,dszie,bsize)
2296     res = fmt.Sprintf("200 PUT done\r\n")
2297     default:
2298         res = fmt.Sprintf("400 What? %v",req)
2299     }
2300     swcc,serr := clnt.Write([]byte(res))
2301     if serr != nil {
2302         fmt.Printf(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v",swcc,serr,res)
2303     }else{
2304         fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2305     }
2306     aconn.Close();
2307     clnt.Close();
2308 }
2309 }
2310 sconn.Close();
2311 }
2312 func (gsh*GshContext)RexecClient(argv[]string)(int,string){
2313     debug := true
2314     Start := time.Now()
2315     if len(argv) == 1 {
2316         return -1,"EmptyARG"
2317     }
2318     argv = argv[1:]
2319     if argv[0] == "-serv" {
2320         gsh.RexecServer(argv[1:])
2321         return 0,"Server"
2322     }
2323     remote := "0.0.0.0:9999"
2324     if argv[0][0] == '@' {
2325         remote = argv[0][1:]
2326         argv = argv[1:]
2327     }
2328     if argv[0] == "-s" {
2329         debug = false
2330         argv = argv[1:]
2331     }
2332     dport, err := net.ResolveTCPAddr("tcp",remote);
2333     if err != nil {
2334         fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
2335         return -1,"AddressError"
2336     }
2337     fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
2338     serv, err := net.DialTCP("tcp",nil,dport)
2339     if err != nil {
2340         fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n",remote,err)
2341         return -1,"CannotConnect"
2342     }
2343     if debug {
2344         al := serv.LocalAddr()
2345         fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
2346     }
2347     reg := ""
2348     res := make([]byte,LINESIZE)
2349     count,err := serv.Read(res)
2350     if err != nil {
2351         fmt.Printf("--En- S: (%d,%v) %v",count,err,string(res))
2352     }
2353     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res)) }
2354
2355     if argv[0] == "GET" {

```

```

2357     savPA := gsh.gshPA
2358     var bsize int = 64*1024
2359     req = fmt.Sprintf("%v\r\n",strings.Join(argv, " "))
2360     fmt.Printf(Elapsed(Start)+"--In- C: %v",req)
2361     fmt.Fprintf(serv,req)
2362     count,err = serv.Read(res)
2363     if err != nil {
2364     }else{
2365         var dsize int64 = 0
2366         var out *os.File = nil
2367         var out_tobeclosed *os.File = nil
2368         var fname string = ""
2369         var rcode int = 0
2370         var pid int = -1
2371         fmt.Sscanf(string(res),"%d %d",&rcode,&dsize)
2372         fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
2373         if 3 <= len(argv) {
2374             fname = argv[2]
2375             if strBegins(fname,"{") {
2376                 xin,xout,err := gsh.Popen(fname,"w")
2377                 if err {
2378                     }else{
2379                         xin.Close()
2380                         defer xout.Close()
2381                         out = xout
2382                         out_tobeclosed = xout
2383                         pid = 0 // should be its pid
2384                     }
2385                 }else{
2386                     // should write to temporary file
2387                     // should suppress ^C on tty
2388                     xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2389                     if err != nil {
2390                         fmt.Println("--En- %v\n",err)
2391                     }
2392                     out = xout
2393                     //fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
2394                 }
2395             }
2396             in,_ := serv.File()
2397             fileRelay("RecvGET",in,out,dsize,bsize)
2398             if 0 <= pid {
2399                 gsh.gshPA = savPA // recovery of Fd(), and more?
2400                 fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
2401                 out_tobeclosed.Close()
2402                 //syscall.Wait4(pid,nil,0,nil) //@@
2403             }
2404         }
2405     }else
2406     if argv[0] == "PUT" {
2407         remote,_ := serv.File()
2408         var local *os.File = nil
2409         var dsize int64 = 32*1024*1024
2410         var bsize int = 64*1024
2411         var ofile string = ""
2412         //fmt.Printf("--I-- Rex %v\n",argv)
2413         if 1 < len(argv) {
2414             fname := argv[1]
2415             if strBegins(fname,"-z") {
2416                 fmt.Sscanf(fname[2:],"%d",&dsize)
2417             }else
2418             if strBegins(fname,"{") {
2419                 xin,xout,err := gsh.Popen(fname,"r")
2420                 if err {
2421                     }else{
2422                         xout.Close()
2423                         defer xin.Close()
2424                         /in = xin
2425                         local = xin
2426                         fmt.Println("--In- [%d] < Upload output of %v\n",
2427                             local.Fd(),fname)
2428                         ofile = "-from."+fname
2429                         dsize = MaxStreamSize
2430                     }
2431                 }else{
2432                     xlocal,err := os.Open(fname)
2433                     if err != nil {
2434                         fmt.Println("--En- (%s)\n",err)
2435                     }
2436                 }else{
2437                     local = xlocal
2438                     fi,_ := local.Stat()
2439                     dsize = fi.Size()
2440                     defer local.Close()
2441                     //fmt.Printf("--I-- Rex in(%v / %v)\n",ofile,dsize)
2442                 }
2443                 ofile = fname
2444                 fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
2445                     fname,dsize,local,err)
2446             }
2447         }
2448         if 2 < len(argv) && argv[2] != "" {
2449             ofile = argv[2]
2450             //fmt.Printf("%d)%v B.ofile=%v\n",len(argv),argv,ofile)
2451         }
2452         //fmt.Printf(Elapsed(Start)+"--I-- Rex out(%v)\n",ofile)
2453         fmt.Printf(Elapsed(Start)+"--In- PUT %v (%v)\n",dsize,bsize)
2454         req = fmt.Sprintf("PUT %v %v (%v)",dsize,ofile)
2455         if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2456         fmt.Fprintf(serv,"%v",req)
2457         count,err = serv.Read(res)
2458         if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
2459         fileRelay("SendPUT",local,remote,dsize,bsize)
2460     }else{
2461         req = fmt.Sprintf("%v\r\n",strings.Join(argv, " "))
2462         if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2463         fmt.Fprintf(serv,"%v",req)
2464         //fmt.Printf("--In- sending RexRequest(%v)\n",len(req))
2465     }
2466     //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
2467     count,err = serv.Read(res)
2468     ress := ""
2469     if count == 0 {
2470         ress = "(nil)\r\n"
2471     }else{
2472         ress = string(res[:count])
2473     }
2474     if err != nil {
2475         fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,ress)
2476     }else{
2477         fmt.Printf(Elapsed(Start)+"--In- S: %v",ress)
2478     }
2479     serv.Close()
2480     //conn.Close()

```

```

2481     var stat string
2482     var rcode int
2483     fmt.Sscanf(res,"%d %s",rcode,&stat)
2484     //fmt.Printf("--D-- Client: %v (%v)",rcode,stat)
2485     return rcode,res
2486   }
2487 }
2488 // <a name="remote-sh">Remote Shell</a>
2489 // gcp file [...] { [host]:[port]:[dir] | dir } // -p | -no-p
2490 func (gsh*GshContext)FileCopy(argv[]string){
2491   var host = ""
2492   var port = ""
2493   var upload = false
2494   var download = false
2495   var xargv = []string{"rex-gcp"}
2496   var srcv = []string{}
2497   var dstv = []string{}
2498   argv = argv[1:]
2499
2500   for _,v := range argv {
2501     /*
2502      if v[0] == '-' { // might be a pseudo file (generated date)
2503        continue
2504      }
2505      */
2506     obj := strings.Split(v,":")
2507     //fmt.Printf("%d %v %v\n",len(obj),v,obj)
2508     if 1 < len(obj) {
2509       host = obj[0]
2510       file := ""
2511       if 0 < len(host) {
2512         gsh.LastServer.host = host
2513       }else{
2514         host = gsh.LastServer.host
2515         port = gsh.LastServer.port
2516       }
2517     }
2518     if 2 < len(obj) {
2519       port = obj[1]
2520       if 0 < len(port) {
2521         gsh.LastServer.port = port
2522       }else{
2523         port = gsh.LastServer.port
2524       }
2525       file = obj[2]
2526     }else{
2527       file = obj[1]
2528     }
2529     if len(srcv) == 0 {
2530       download = true
2531       srcv = append(srcv,file)
2532       continue
2533     }
2534     upload = true
2535     dstv = append(dstv,file)
2536     continue
2537   }
2538   /*
2539   idx := strings.Index(v,:")
2540   if 0 <= idx {
2541     remote = v[0:idx]
2542     if len(srcv) == 0 {
2543       download = true
2544       srcv = append(srcv,v[idx+1:])
2545       continue
2546     }
2547     upload = true
2548     dstv = append(dstv,v[idx+1:])
2549     continue
2550   }
2551   */
2552   if download {
2553     dstv = append(dstv,v)
2554   }else{
2555     srcv = append(srcv,v)
2556   }
2557 }
2558 hostport := "@" + host + ":" + port
2559 if upload {
2560   if host != "" { xargv = append(xargv,hostport) }
2561   xargv = append(xargv,"PUT")
2562   xargv = append(xargv,srcv[0:...])
2563   xargv = append(xargv,dstv[0:...])
2564   //fmt.Printf("--I-- FileCopy PUT gsh://%s/%v < %v // %v\n",hostport,dstv,srcv,xargv)
2565   fmt.Println("--I-- FileCopy PUT gsh://%s/%v < %v\n",hostport,dstv,srcv)
2566   gsh.RexecClient(xargv)
2567 }else{
2568   if download {
2569     if host != "" { xargv = append(xargv,hostport) }
2570     xargv = append(xargv,"GET")
2571     xargv = append(xargv,srcv[0:...])
2572     xargv = append(xargv,dstv[0:...])
2573   //fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v // %v\n",hostport,srcv,dstv,xargv)
2574   fmt.Println("--I-- FileCopy GET gsh://%v/%v > %v\n",hostport,srcv,dstv)
2575   gsh.RexecClient(xargv)
2576 }else{
2577 }
2578 }
2579
2580 // target
2581 func (gsh*GshContext)Treelpath(rloc string)(string){
2582   cwd, _ := os.Getwd()
2583   os.Chdir(gsh.RWD)
2584   os.Chdir(rloc)
2585   twd, _ := os.Getwd()
2586   os.Chdir(cwd)
2587
2588   tpath := twd + "/" + rloc
2589   return tpath
2590 }
2591 // join to remote GShell - [user@host[:port] or cd host:[port]:path
2592 func (gsh*GshContext)Rjoin(argv[]string){
2593   if len(argv) <= 1 {
2594     fmt.Printf("--I-- current server = %v\n",gsh.RSERV)
2595     return
2596   }
2597   serv := argv[1]
2598   servv := strings.Split(servv,":")
2599   if 1 < len(servv) {
2600     if servv[0] == "lo" {
2601       servv[0] = "localhost"
2602     }
2603   }
2604   switch len(servv) {

```

```

2605     case 1:
2606         //if strings.Index(serv,":") < 0 {
2607             serv = serv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
2608         //}
2609     case 2: // host:port
2610         serv = strings.Join(servv,:")
2611     }
2612     xargv := []string{"rex-join","@"+serv,"HELO"}
2613     rcode,stat := gsh.RexecClient(xargv)
2614     if (rcode / 100) == 2 {
2615         fmt.Printf("--I-- OK Joined (%v) %v\n",rcode,stat)
2616         gsh.RSERV = serv
2617     }else{
2618         fmt.Printf("--I-- NG, could not joined (%v) %v\n",rcode,stat)
2619     }
2620 }
2621 func (gsh*GshContext)Rexec(argv[]string){
2622     if len(argv) <= 1 {
2623         fmt.Printf("--I-- rexec command [ | {file} || {command} ]\n",gsh.RSERV)
2624         return
2625     }
2626     /*
2627     nargv := gshScanArg(strings.Join(argv," "),0)
2628     fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
2629     if nargv[1][0] != '{' {
2630         nargv[1] = "(" + nargv[1] + ")"
2631         fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
2632     }
2633     argv = nargv
2634     */
2635     argv := []string{}
2636     argv = append(argv,""+strings.Join(argv[1:]," ")+"")
2637     fmt.Printf("--D-- nargc=%d %v\n",len(argv),argv)
2638     argv = argv
2639
2640     xargv := []string{"rex-exec","@"+gsh.RSERV,"GET"}
2641     argv = append(xargv,argv...)
2642     argv = append(argv,"/dev/tty")
2643     rcode,stat := gsh.RexecClient(xargv)
2644     if (rcode / 100) == 2 {
2645         fmt.Printf("--I-- OK Rexec (%v) %v\n",rcode,stat)
2646     }else{
2647         fmt.Printf("--I-- NG Rexec (%v) %v\n",rcode,stat)
2648     }
2649 }
2650 func (gsh*GshContext)Rchdir(argv[]string){
2651     if len(argv) <= 1 {
2652         return
2653     }
2654     cwd, _ := os.Getwd()
2655     os.Chdir(gsh.RWD)
2656     os.Chdir(argv[1])
2657     twd, _ := os.Getwd()
2658     gsh.RWD = twd
2659     fmt.Printf("--I-- JWD=%v\n",twd)
2660     os.Chdir(cwd)
2661 }
2662 func (gsh*GshContext)Rpwd(argv[]string){
2663     fmt.Println("%v\n",gsh.RWD)
2664 }
2665 func (gsh*GshContext)Rls(argv[]string){
2666     cwd, _ := os.Getwd()
2667     os.Chdir(gsh.RWD)
2668     argv[0] = "ls"
2669     gsh.xfind(argv)
2670     os.Chdir(cwd)
2671 }
2672 func (gsh*GshContext)Rput(argv[]string){
2673     var local string = ""
2674     var remote string = ""
2675     if 1 < len(argv) {
2676         local = argv[1]
2677         remote = local // base name
2678     }
2679     if 2 < len(argv) {
2680         remote = argv[2]
2681     }
2682     fmt.Printf("--I-- jput from=%v to=%v\n",local,gsh.Trepath(remote))
2683 }
2684 func (gsh*GshContext)Rget(argv[]string){
2685     var remote string = ""
2686     var local string = ""
2687     if 1 < len(argv) {
2688         remote = argv[1]
2689         local = remote // base name
2690     }
2691     if 2 < len(argv) {
2692         local = argv[2]
2693     }
2694     fmt.Printf("--I-- jget from=%v to=%v\n",gsh.Trepath(remote),local)
2695 }
2696 }
2697 // <a name="network">network</a>
2698 // -s, -si, -so // bi-directional, source, sync (maybe socket)
2699 func (gshCtx*GshContext)sconnect(inTCP bool, argv []string) {
2700     gshPA := gshCtx.gshPA
2701     if len(argv) < 2 {
2702         fmt.Printf("Usage: -s [host]:[port[:udp]]\n")
2703         return
2704     }
2705     remote := argv[1]
2706     if remote == ":" { remote = "0.0.0.0:9999" }
2707
2708     if inTCP { // TCP
2709         dport, err := net.ResolveTCPAddr("tcp",remote);
2710         if err != nil {
2711             fmt.Printf("Address error: %s (%s)\n",remote,err)
2712             return
2713         }
2714         conn, err := net.DialTCP("tcp",nil,dport)
2715         if err != nil {
2716             fmt.Printf("Connection error: %s (%s)\n",remote,err)
2717             return
2718         }
2719         file, _ := conn.File();
2720         fd := file.Fd()
2721         fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,fd)
2722
2723         savfd := gshPA.Files[1]
2724         gshPA.Files[1] = fd;
2725         gshCtx.gshellv(argv[2:])
2726         gshPA.Files[1] = savfd
2727         file.Close()
2728 }
```

```

2729     conn.Close()
2730 }else{
2731     //dport, err := net.ResolveUDPAddr("udp4",remote);
2732     dport, err := net.ResolveUDPAddr("udp",remote);
2733     if err != nil {
2734         fmt.Printf("Address error: %s (%s)\n",remote,err)
2735         return
2736     }
2737     //conn, err := net.DialUDP("udp4",nil,dport)
2738     conn, err := net.DialUDP("udp",nil,dport)
2739     if err != nil {
2740         fmt.Printf("Connection error: %s (%s)\n",remote,err)
2741         return
2742     }
2743     file, _ := conn.File();
2744     fd := file.Fd()
2745
2746     ar := conn.RemoteAddr()
2747     //al := conn.LocalAddr()
2748     fmt.Println('Socket: connected to %s [%s], socket[%d]\n',
2749                 remote,ar.String(),fd)
2750
2751     savfd := gshPA.Files[1]
2752     gshPA.Files[1] = fd;
2753     gshCtx.gshellv(argv[2:])
2754     gshPA.Files[1] = savfd
2755     file.Close()
2756     conn.Close()
2757 }
2758 }
2759 func (gshCtx*GshContext)accept(inTCP bool, argv []string) {
2760     gshPA := gshCtx.gshPA
2761     if len(argv) < 2 {
2762         fmt.Printf("Usage: -ac [host]:[port[.udp]]\n")
2763         return
2764     }
2765     local := argv[1]
2766     if local == ":" { local = "0.0.0.0:9999" }
2767     if inTCP { // TCP
2768         port, err := net.ResolveTCPAddr("tcp",local);
2769         if err != nil {
2770             fmt.Printf("Address error: %s (%s)\n",local,err)
2771             return
2772         }
2773         //fmt.Printf("Listen at %s...\n",local);
2774         sconn, err := net.ListenTCP("tcp", port)
2775         if err != nil {
2776             fmt.Printf("Listen error: %s (%s)\n",local,err)
2777             return
2778         }
2779         //fmt.Printf("Accepting at %s...\n",local);
2780         aconn, err := sconn.AcceptTCP()
2781         if err != nil {
2782             fmt.Printf("Accept error: %s (%s)\n",local,err)
2783             return
2784         }
2785         file, _ := aconn.File()
2786         fd := file.Fd()
2787         fmt.Printf("Accepted TCP at %s [%d]\n",local,fd)
2788
2789         savfd := gshPA.Files[0]
2790         gshPA.Files[0] = fd;
2791         gshCtx.gshellv(argv[2:])
2792         gshPA.Files[0] = savfd
2793
2794         sconn.Close();
2795         aconn.Close();
2796         file.Close();
2797     }else{
2798         //port, err := net.ResolveUDPAddr("udp4",local);
2799         port, err := net.ResolveUDPAddr("udp",local);
2800         if err != nil {
2801             fmt.Printf("Address error: %s (%s)\n",local,err)
2802             return
2803         }
2804         fmt.Printf("Listen UDP at %s...\n",local);
2805         //uconn, err := net.ListenUDP("udp4", port)
2806         uconn, err := net.ListenUDP("udp", port)
2807         if err != nil {
2808             fmt.Printf("Listen error: %s (%s)\n",local,err)
2809             return
2810         }
2811         file, _ := uconn.File()
2812         fd := file.Fd()
2813         ar := uconn.RemoteAddr()
2814         remote :=
2815         if ar != nil { remote = ar.String() }
2816         if remote == "" { remote = "?" }
2817
2818         // not yet received
2819         //fmt.Printf("Accepted at %s [%d] <- %s\n",local,fd,"")
2820
2821         savfd := gshPA.Files[0]
2822         gshPA.Files[0] = fd;
2823         savenv := gshPA.Env
2824         gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
2825         gshCtx.gshellv(argv[2:])
2826         gshPA.Env = savenv
2827         gshPA.Files[0] = savfd
2828
2829         uconn.Close();
2830         file.Close();
2831     }
2832 }
2833
2834 // empty line command
2835 func (gshCtx*GshContext)xPwd(argv[]string){
2836     // execute context command, pwd + date
2837     // context notation, representation scheme, to be resumed at re-login
2838     cwd, _ := os.Getwd()
2839     switch {
2840     case isn("-a",argv):
2841         gshCtx.ShowChdirHistory(argv)
2842     case isn("-ls",argv):
2843         showFileInfo(cwd,argv)
2844     default:
2845         fmt.Println("%s\n",cwd)
2846     case isn("-v",argv): // obsolete empty command
2847         t := time.Now()
2848         date := t.Format(time.UnixDate)
2849         exe, _ := os.Executable()
2850         host, _ := os.Hostname()
2851         fmt.Printf("{PWD=%s\"", cwd)
2852         fmt.Printf(" HOST=%s\"", host)
2853     }
2854 }
```

```

2853     fmt.Printf(" DATE=%s\",date)
2854     fmt.Printf(" TIME=%s\",t.String())
2855     fmt.Printf(" PID=%d\",os.Getpid())
2856     fmt.Printf(" EXE=%s\",exe)
2857     fmt.Println("\n")
2858 }
2859 }
2860
2861 // <a name="history">History</a>
2862 // these should be browsed and edited by HTTP browser
2863 // show the time of command with -t and directory with -ls
2864 // openfile-history, sort by -a -m -c
2865 // sort by elapsed time by -t -s
2866 // search by "more" like interface
2867 // edit history
2868 // sort history, and wc or uniq
2869 // CPU and other resource consumptions
2870 // limit showing range (by time or so)
2871 // export / import history
2872 func (gshCtx *GshContext)xHistory(argv []string){
2873     atWorkDirX := -1
2874     if 1 < len(argv) && strBegins(argv[1],"@") {
2875         atWorkDirX,_ = strconv.Atoi(argv[1][1:])
2876     }
2877     //fmt.Printf("--D-- showHistory(%v)\n",argv)
2878     for i, v := range gshCtx.CommandHistory {
2879         // exclude commands not to be listed by default
2880         // internal commands may be suppressed by default
2881         if v.CmdLine == "" && !isin("-a",argv) {
2882             continue;
2883         }
2884         if 0 <= atWorkDirX {
2885             if v.WorkDirX != atWorkDirX {
2886                 continue
2887             }
2888         if !isin("-n",argv){ // like "fc"
2889             fmt.Printf("!%-2d ",i)
2890         }
2891         if isin("-v",argv){
2892             fmt.Println(v) // should be with it date
2893         }else{
2894             if isin("-l",argv) || isin("-10",argv) {
2895                 elps := v.EndAt.Sub(v.StartAt);
2896                 start := v.StartAt.Format(time.Stamp)
2897                 fmt.Printf("%d %v.%v",v.WorkDirX,
2898                             fmt.Sprintf("%v %11v/t",start,elps))
2899             }
2900             if isin("-1",argv) && !isin("-10",argv){
2901                 fmt.Printf("%v",Rusagef("%t %t// %s",argv,v.Rusagev))
2902             }
2903             if isin("-at",argv) { // isin("-ls",argv){
2904                 dhi := v.WorkDirX // workdir history index
2905                 fmt.Printf("%d %v",dhi,v.WorkDir)
2906                 // show the FileInfo of the output command??
2907             }
2908             fmt.Printf("%s",v.CmdLine)
2909             fmt.Printf("\n")
2910         }
2911     }
2912 }
2913 }
2914 // !n - history index
2915 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
2916     if gline[0] == '!' {
2917         hix, err := strconv.Atoi(gline[1:])
2918         if err != nil {
2919             fmt.Printf("--E-- (%s : range)\n",hix)
2920             return "", false, true
2921         }
2922         if hix < 0 || len(gshCtx.CommandHistory) <= hix {
2923             fmt.Printf("--E-- (%d : out of range)\n",hix)
2924             return "", false, true
2925         }
2926         return gshCtx.CommandHistory[hix].CmdLine, false, false
2927     }
2928     // search
2929     //for i, v := range gshCtx.CommandHistory {
2930     //}
2931     return gline, false, false
2932 }
2933 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
2934     if 0 <= hix && hix < len(gsh.CommandHistory) {
2935         return gsh.CommandHistory[hix].CmdLine,true
2936     }
2937     return "",false
2938 }
2939
2940 // temporary adding to PATH environment
2941 // cd name -lib for LD_LIBRARY_PATH
2942 // chdir with directory history (date + full-path)
2943 // -s for sort option (by visit date or so)
2944 func (gsh*GshContext>ShowChdirHistory(i int,v GChdirHistory, argv []string){
2945     fmt.Printf("%-2d ",v.CmdIndex) // the first command at this WorkDir
2946     fmt.Println("@%d ",i)
2947     fmt.Printf("[%v] ",v.MovedAt.Format(time.Stamp))
2948     showFileInfo(v.Dir,argv)
2949 }
2950 func (gsh*GshContext>ShowChdirHistory(argv []string){
2951     for i, v := range gsh.ChdirHistory {
2952         gsh.ShowChdirHistory(i,v,argv)
2953     }
2954 }
2955 func skipOpts(argv[]string)(int){
2956     for i,v := range argv {
2957         if strBegins(v,"-") {
2958             }else{
2959                 return i
2960             }
2961         }
2962     return -1
2963 }
2964 func (gshCtx*GshContext)xChdir(argv []string){
2965     cdhist := gshCtx.ChdirHistory
2966     if isin("?",argv) || isin("-t",argv) || isin("-a",argv) {
2967         gshCtx.ShowChdirHistory(argv)
2968         return
2969     }
2970     pwd, _ := os.Getwd()
2971     dir := ""
2972     if len(argv) <= 1 {
2973         dir = toFullPath("-")
2974     }else{
2975         i := skipopts(argv[1:])
2976         if i < 0 {

```

```

2977     dir = toFullPath("~")
2978 }else{
2979     dir = argv[1+i]
2980 }
2981 }if strBegins(dir,"~"){
2982     if dir == "@0" { // obsolete
2983         dir = gshCtx.StartDir
2984     }else{
2985         if dir == "@"{
2986             index := len(cdhist) - 1
2987             if 0 < index { index -= 1 }
2988             dir = cdhist[index].Dir
2989         }else{
2990             index, err := strconv.Atoi(argv[1:])
2991             if err != nil {
2992                 fmt.Printf("--E-- xChdir(%v)\n",err)
2993                 dir = "?"
2994             }else{
2995                 if len(gshCtx.ChdirHistory) <= index {
2996                     fmt.Printf("--E-- xChdir(history range error)\n")
2997                     dir = "?"
2998                 }else{
2999                     dir = cdhist[index].Dir
3000                 }
3001             }
3002         }
3003     }if dir != "?"{
3004         err := os.Ckdir(dir)
3005         if err != nil {
3006             fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
3007         }else{
3008             cwd, _ := os.Getwd()
3009             if cwd != pwd {
3010                 hist1 := GCkdirHistory {}
3011                 hist1.Dir = cwd
3012                 hist1.MovedAt = time.Now()
3013                 hist1.CmdIndex = len(gshCtx.CommandHistory)+1
3014                 gshCtx.ChdirHistory = append(cdhist,hist1)
3015                 if !isin("-s",argv){
3016                     cwd, _ := os.Getwd()
3017                     fmt.Printf("%s\n",cwd)
3018                     ix := len(gshCtx.ChdirHistory)-1
3019                     gshCtx.ShowChdirHistory1(ix,hist1,argv)
3020                 }
3021             }
3022         }
3023     }
3024 }if isin("-ls",argv){
3025     cwd, _ := os.Getwd()
3026     showFileInfo(cwd,argv);
3027 }
3028 }
3029 }
3030 func TimeValSub(tv1 *syscall.Timeval, tv2 *syscall.Timeval){
3031     *tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
3032 }
3033 func RusageSub(r1, r2 [2]syscall.Rusage)([2]syscall.Rusage){
3034     TimeValSub(&r1[0].Utime,&r2[0].Utime)
3035     TimeValSub(&r1[0].Stime,&r2[0].Stime)
3036     TimeValSub(&r1[1].Utime,&r2[1].Utime)
3037     TimeValSub(&r1[1].Stime,&r2[1].Stime)
3038     return r1
3039 }
3040 func TimeValAdd(tv1 syscall.Timeval, tv2 syscall.Timeval)(syscall.Timeval){
3041     tvs := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
3042     return tvs
3043 }
3044 */
3045 func RusageAddv(r1, r2 [2]syscall.Rusage)([2]syscall.Rusage){
3046     TimeValAdd(r1[0].Utime,r2[0].Utime)
3047     TimeValAdd(r1[0].Stime,r2[0].Stime)
3048     TimeValAdd(r1[1].Utime,r2[1].Utime)
3049     TimeValAdd(r1[1].Stime,r2[1].Stime)
3050     return r1
3051 }
3052 */
3053
3054 // <a name="rusage">Resource Usage</a>
3055 func sRusagef(fmtSpec string, argv []string, ru [2]syscall.Rusage)(string){
3056     // ru[0] self , ru[1] children
3057     ut := TimeValadd(ru[0].Utime,ru[1].Utime)
3058     st := TimeValadd(ru[0].Stime,ru[1].Stime)
3059     uu := (ut.Sec*1000000 + int64(ut.Usec)) * 1000
3060     su := (st.Sec*1000000 + int64(st.Usec)) * 1000
3061     tu := uu + su
3062     ret := fmt.Sprintf("%v/sum",abftime(tu))
3063     ret += fmt.Sprintf(", %v/usr",abftime(uu))
3064     ret += fmt.Sprintf(", %v/sys",abftime(su))
3065     return ret
3066 }
3067 func Rusagef(fmtSpec string, argv []string, ru [2]syscall.Rusage)(string){
3068     ut := TimeValadd(ru[0].Utime,ru[1].Utime)
3069     st := TimeValadd(ru[0].Stime,ru[1].Stime)
3070     fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) // /ru[1].Utime.Sec,ru[1].Utime.Usec)
3071     fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) // /ru[1].Stime.Sec,ru[1].Stime.Usec)
3072     return ""
3073 }
3074 func Getrusagev(([2]syscall.Rusage){
3075     var rvv = [2]syscall.Rusage{
3076         syscall.Getrusage(syscall.RUSAGE_SELF,&rvv[0])
3077         syscall.Getrusage(syscall.RUSAGE_CHILDREN,&rvv[1])
3078     }
3079 }
3080 func showRusage(what string,argv []string, ru *syscall.Rusage){
3081     fmt.Printf(":@: %s",what);
3082     fmt.Printf("User=%d.%06ds",ru.Utime.Sec,ru.Utime.Usec)
3083     fmt.Printf(" Sys=%d.%06ds",ru.Stime.Sec,ru.Stime.Usec)
3084     fmt.Printf(" Rss=%vB",ru.Maxrss)
3085     if isin("-l",argv) {
3086         fmt.Printf(" MinFlt=%v",ru.Minflt)
3087         fmt.Printf(" MajFlt=%v",ru.Majflt)
3088         fmt.Printf(" IXRSS=%vB",ru.Ixrss)
3089         fmt.Printf(" IDRSS=%vB",ru.Idrss)
3090         fmt.Printf(" Nswap=%vB",ru.Nswap)
3091         fmt.Printf(" Read=%v",ru.Inblock)
3092         fmt.Printf(" Write=%v",ru.Outblock)
3093     }
3094     fmt.Printf(" Snd=%v",ru.Msgsnd)
3095     fmt.Printf(" Rcv=%v",ru.Msgrcv)
3096     //if isin("-l",argv) {
3097         fmt.Printf(" Sig=%v",ru.Nsignals)
3098     //}
3099     fmt.Printf("\n");
3100 }

```

```

3101 func (gshCtx *GshContext)xTime(argv[]string)(bool){
3102     if 2 < len(argv){
3103         gshCtx.LastRusage = syscall.Rusage{}
3104         rusage := Getrusagev()
3105         fin := gshCtx.gshellv(argv[1:])
3106         rusagev1 := Getrusagev()
3107         showUsage(argv[1], argv, &gshCtx.LastRusage)
3108         rusage := RusageSubv(rusagev2, rusagev1)
3109         showUsage("self", argv, &rusagev[0])
3110         showUsage("chld", argv, &rusagev[1])
3111         return fin
3112     }else{
3113         rusage:= syscall.Rusage {}
3114         syscall.Getrusage(syscall.RUSAGE_SELF,&rusage)
3115         showUsage("self", argv, &rusage)
3116         syscall.Getrusage(syscall.RUSAGE_CHILDREN,&rusage)
3117         showUsage("chld", argv, &rusage)
3118         return false
3119     }
3120 }
3121 func (gshCtx *GshContext)xJobs(argv[]string){
3122     fmt.Printf("--# Jobs\n",len(gshCtx.BackGroundJobs))
3123     for ji, pid := range gshCtx.BackGroundJobs {
3124         //wstat := syscall.WaitStatus {0}
3125         rusage := syscall.Rusage {}
3126         //wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&rusage);
3127         wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
3128         if err != nil {
3129             fmt.Printf("--E-- %d (%v)\n",ji,pid,err)
3130         }else{
3131             fmt.Printf("%d(%d)\n",ji,pid,wpid)
3132             showRusage("chld",argv,&rusage)
3133         }
3134     }
3135 }
3136 func (gsh*GshContext)inBackground(argv[]string)(bool){
3137     if gsh.CmdTrace { fmt.Println("--I-- inBackground(%v)\n",argv) }
3138     gsh.BackGround = true // set background option
3139     xfin := false
3140     xfin = gsh.gshellv(argv)
3141     gsh.BackGround = false
3142     return xfin
3143 }
3144 // -o file without command means just opening it and refer by #N
3145 // should be listed by "files" command
3146 func (gshCtx*GshContext)xOpen(argv[]string){
3147     var pv = []int{-1,-1}
3148     err := syscall.Pipe(pv)
3149     fmt.Printf("--I-- pipe()=%d,%d(%v)\n",pv[0],pv[1],err)
3150 }
3151 func (gshCtx*GshContext)fromPipe(argv[]string){
3152 }
3153 func (gshCtx*GshContext)xClose(argv[]string){
3154 }
3155
3156 // <a name="redirect">redirect</a>
3157 func (gshCtx*GshContext)redirect(argv[]string)(bool){
3158     if len(argv) < 2 {
3159         return false
3160     }
3161     cmd := argv[0]
3162     fname := argv[1]
3163     var file *os.File = nil
3164     ffdix := 0
3165     mode := os.O_RDONLY
3166     switch {
3167     case cmd == "-i" || cmd == "<":
3168         ffdix = 0
3169         mode = os.O_RDONLY
3170     case cmd == "-o" || cmd == ">":
3171         ffdix = 1
3172         mode = os.O_RDWR | os.O_CREATE
3173     case cmd == "-a" || cmd == ">>":
3174         ffdix = 1
3175         mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3176     }
3177     if fname[0] == '#' {
3178         fd, err := strconv.Atoi(fname[1:])
3179         if err != nil {
3180             fmt.Printf("--E-- (%v)\n",err)
3181             return false
3182         }
3183         file = os.NewFile(uintptr(fd), "MaybePipe")
3184     }else{
3185         xfile, err := os.OpenFile(argv[1], mode, 0600)
3186         if err != nil {
3187             fmt.Printf("--E-- (%s)\n",err)
3188             return false
3189         }
3190         file = xfile
3191     }
3192     gshPA := gshCtx.gshPA
3193     savfd := gshPA.Files[ffdix]
3194     gshPA.Files[ffdix] = file.Fd()
3195     fmt.Printf("--I-- Opened %d %s\n",file.Fd(),argv[1])
3196     gshCtx.gshellv(argv[2:])
3197     gshPA.Files[ffdix] = savfd
3198
3199
3200     return false
3201 }
3202
3203 }
3204
3205 //fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
3206 func httpHandler(res http.ResponseWriter, req *http.Request){
3207     path := req.URL.Path
3208     fmt.Printf("--I-- Got HTTP Request(%s)\n",path)
3209     {
3210         gshCtxBuf, _ := setupGshContext()
3211         gshCtx :=&gshCtxBuf
3212         fmt.Printf("--I-- %s\n",path[1:])
3213         gshCtx.tgshell1(path[1:])
3214     }
3215     fmt.Fprintf(res, "Hello(^~)//\n%s\n",path)
3216 }
3217 func (gshCtx *GshContext) httpServer(argv []string){
3218     http.HandleFunc("/", httpHandler)
3219     accport := "localhost:9999"
3220     fmt.Println("--I-- HTTP Server Start at [%s]\n",accport)
3221     http.ListenAndServe(accport,nil)
3222 }
3223 func (gshCtx *GshContext)xGo(argv[]string){
3224     go gshCtx.gshellv(argv[1:]);

```

```

3225 }
3226 func (gshCtx *GshContext) xPs(argv[]string)(){
3227 }
3228
3229 // <a name="plugin">Plugin</a>
3230 // plugin [-ls [names]] to list plugins
3231 // Reference: <a href="https://golang.org/src/plugin/">plugin</a> source code
3232 func (gshCtx *GshContext) whichPlugin(name string,argv[]string)(pi *PluginInfo){
3233     pi = nil
3234     for _,p := range gshCtx.PluginFuncs {
3235         if p.Name == name && pi == nil {
3236             pi = &p
3237         }
3238         if !isin("-s",argv){
3239             //fmt.Printf("%v %v ",i,p)
3240             if isin("-ls",argv){
3241                 showFileInfo(p.Path,argv)
3242             }else{
3243                 fmt.Printf("%s\n",p.Name)
3244             }
3245         }
3246     }
3247     return pi
3248 }
3249 func (gshCtx *GshContext) xPlugin(argv[]string) (error) {
3250     if len(argv) == 0 || argv[0] == "-ls" {
3251         gshCtx.whichPlugin("",argv)
3252         return nil
3253     }
3254     name := argv[0]
3255     Pin := gshCtx.whichPlugin(name,[]string("-s"))
3256     if Pin != nil {
3257         os.Args = argv // should be recovered?
3258         Pin.Addr().func }()
3259         return nil
3260     }
3261     sofile := toFullPath(argv[0] + ".so") // or find it by which($PATH)
3262
3263     p, err := plugin.Open(sofile)
3264     if err != nil {
3265         fmt.Printf("--E-- plugin.Open(%s)(%v)\n", sofile, err)
3266         return err
3267     }
3268     fname := "Main"
3269     f, err := p.Lookup(fname)
3270     if( err != nil ){
3271         fmt.Printf("--E-- plugin.Lookup(%s)(%v)\n", fname, err)
3272         return err
3273     }
3274     pin := PluginInfo {p,f,name,sofile}
3275     gshCtx.PluginFuncs = append(gshCtx.PluginFuncs,pin)
3276     fmt.Printf("--I-- added (%d)\n",len(gshCtx.PluginFuncs))
3277
3278     //fmt.Printf("--I-- first call(%s:%s)%v\n", sofile, fname, argv)
3279     os.Args = argv
3280     f.func }()
3281     return err
3282 }
3283 func (gshCtx*GshContext) Args(argv[]string){
3284     for i,v := range os.Args {
3285         fmt.Printf("[%v] %v\n",i,v)
3286     }
3287 }
3288 func (gshCtx *GshContext) showVersion(argv[]string){
3289     if isin("-l",argv) {
3290         fmt.Printf("%v%v (%v)",NAME,VERSION,DATE);
3291     }else{
3292         fmt.Printf("%v",VERSION);
3293     }
3294     if isin("-a",argv) {
3295         fmt.Printf(" %s",AUTHOR)
3296     }
3297     if !isin("-n",argv) {
3298         fmt.Printf("\n")
3299     }
3300 }
3301
3302 // <a name="scanf">Scanf</a> // string decomposer
3303 // scanf [format] [input]
3304 func scanf(sstr string)(strv[]string){
3305     strv = strings.Split(sstr," ")
3306     return strv
3307 }
3308 func scanUntil(src,end string)(rstr string,leng int){
3309     idx := strings.Index(src,end)
3310     if 0 <= idx {
3311         rstr = src[0:idx]
3312         return rstr,idx+leng(end)
3313     }
3314     return src,0
3315 }
3316
3317 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
3318 func (gsh*GshContext) printVal(fmts string, vstr string, optv[]string){
3319     //vint,err := strconv.Atoi(vstr)
3320     var ival int64 = 0
3321     n := 0
3322     err := error(nil)
3323     if strBegins(vstr,"_") {
3324         vx,_ := strconv.Atoi(vstr[1:])
3325         if vx < len(gsh.iValues) {
3326             vstr = gsh.iValues[vx]
3327         }else{
3328         }
3329     } // should use Eval()
3330     if strBegins(vstr,"0x") {
3331         n,err = fmt.Sscanf(vstr[2:], "%x", &ival)
3332     }else{
3333         n,err = fmt.Sscanf(vstr,"%d", &ival)
3334     }
3335     //fmt.Printf("--D-- n=%d err=%(v) {ts}=%v\n",n,err,vstr, ival)
3336     }
3337     if n == 1 && err == nil {
3338         //fmt.Printf("--D-- formatn(%v) ival(%v)\n",fmts,ival)
3339         fmt.Printf("%"+fmts,ival)
3340     }else{
3341         if isin("-bn",optv){
3342             fmt.Printf("%"+fmts,filepath.Base(vstr))
3343         }else{
3344             fmt.Printf("%"+fmts,vstr)
3345         }
3346     }
3347 }
3348 func (gsh*GshContext) printfv(fmts,div string,argv[]string,optv[]string,list[]string){

```

```

3349     //fmt.Printf("%d",len(list))
3350     //curfmt := "v"
3351     outlen := 0
3352     curfmt := gsh.iFormat
3353
3354     if 0 < len(fmts) {
3355         for xi := 0; xi < len(fmts); xi++ {
3356             fch := fmts[xi]
3357             if fch == '%' {
3358                 if xi+1 < len(fmts) {
3359                     curfmt = string(fmts[xi+1])
3360                 gsh.iFormat = curfmt
3361                 xi += 1
3362                 if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3363                     vals,leng := scanUntil(fmts[xi+1:],")")
3364                     //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,leng)
3365                     gsh.printVal(curfmt,vals,optv)
3366                     xi += 2+leng-1
3367                     outlen += 1
3368                 }
3369                 continue
3370             }
3371             if fch == '_' {
3372                 hi,leng := scanInt(fmts[xi+1:])
3373                 if 0 < leng {
3374                     if hi < len(gsh.iValues) {
3375                         gsh.printVal(curfmt,gsh.iValues[hi],optv)
3376                         outlen += 1 // should be the real length
3377                     }else{
3378                         fmt.Printf("(out-range)")
3379                     }
3380                     xi += leng
3381                     continue;
3382                 }
3383             }
3384             fmt.Printf("%c",fch)
3385             outlen += 1
3386         }
3387     }else{
3388         //fmt.Printf("--D-- print %s\n")
3389         for i,v := range list {
3390             if 0 < i {
3391                 fmt.Printf(div)
3392             }
3393             gsh.printVal(curfmt,v,optv)
3394             outlen += 1
3395         }
3396     }
3397     if 0 < outlen {
3398         fmt.Printf("\n")
3399     }
3400 }
3401 }
3402 func (gsh*GshContext)Scany(argv[]string){
3403     //fmt.Printf("--D-- Scanv(%v)\n",argv)
3404     if len(argv) == 1 {
3405         return
3406     }
3407     argv = argv[1:]
3408     fmts := ""
3409     if strBegins(argv[0],"-F") {
3410         fmts = argv[0]
3411         gsh.iDelimiter = fmts
3412         argv = argv[1:]
3413     }
3414     input := strings.Join(argv," ")
3415     if fmts == "" { // simple decomposition
3416         v := scanv(input)
3417         gsh.iValues = v
3418         //fmt.Printf("%v\n",strings.Join(v,","))
3419     }else{
3420         v := make([]string,8)
3421         n,err := fmt.Sscanf(input,fmts,&v[0],&v[1],&v[2],&v[3])
3422         fmt.Printf("--D-- Scanf ->(%v) n=%d err=(%v)\n",v,n,err)
3423         gsh.iValues = v
3424     }
3425 }
3426 func (gsh*GshContext)Printv(argv[]string){
3427     if false { /*@U
3428         fmt.Printf("%v\n",strings.Join(argv[1:]," "))
3429     return
3430     */
3431     //fmt.Printf("--D-- Printv(%v)\n",argv)
3432     //fmt.Printf("%v\n",strings.Join(gsh.iValues,""))
3433     div := gsh.iDelimiter
3434     fmts := ""
3435     argv = argv[1:]
3436     if 0 < len(argv) {
3437         if strBegins(argv[0],"-F") {
3438             div = argv[0][2:]
3439             argv = argv[1:]
3440         }
3441     }
3442     optv := []string{}
3443     for _,v := range argv {
3444         if strBegins(v,"_"){
3445             optv = append(optv,v)
3446             argv = argv[1:]
3447         }else{
3448             break;
3449         }
3450     }
3451     if 0 < len(argv) {
3452         fmts = strings.Join(argv," ")
3453     }
3454     gsh.printfv(fmts,div,argv,optv,gsh.iValues)
3455 }
3456 func (gsh*GshContext)Basename(argv[]string){
3457     for i,v := range gsh.iValues {
3458         gsh.iValues[i] = filepath.Base(v)
3459     }
3460 }
3461 func (gsh*GshContext)Sortv(argv[]string){
3462     sv := gsh.iValues
3463     sort.Slice(sv , func(i,j int) bool {
3464         return sv[i] < sv[j]
3465     })
3466 }
3467 }
3468 func (gsh*GshContext)Shiftv(argv[]string){
3469     vi := len(gsh.iValues)
3470     if 0 < vi {
3471         if isin("r",argv) {
3472             top := gsh.iValues[0]

```

```

3473     gsh.iValues = append(gsh.iValues[1:],top)
3474   }else{
3475     gsh.iValues = gsh.iValues[1:]
3476   }
3477 }
3478 }
3479 func (gsh*GshContext)Enq(argv[]string){
3480 }
3481 func (gsh*GshContext)Deq(argv[]string){
3482 }
3483 func (gsh*GshContext)Push(argv[]string){
3484   gsh.iValStack = append(gsh.iValStack,argv[1:])
3485   fmt.Printf("depth=%d\n",len(gsh.iValStack))
3486 }
3487 func (gsh*GshContext)Dump(argv[]string){
3488   for i,v := range gsh.iValStack {
3489     fmt.Printf("%d %v\n",i,v)
3490   }
3491 }
3492 func (gsh*GshContext)Pop(argv[]string){
3493   depth := len(gsh.iValStack)
3494   if 0 < depth {
3495     v := gsh.iValStack[depth-1]
3496     if isin(".cat",argv){
3497       gsh.iValues = append(gsh.iValues,v...)
3498     }else{
3499       gsh.iValues = v
3500     }
3501     gsh.iValStack = gsh.iValStack[0:depth-1]
3502     fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
3503   }else{
3504     fmt.Printf("depth=%d\n",depth)
3505   }
3506 }
3507 }
3508 // <a name="interpreter">Command Interpreter</a>
3509 func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
3510   fin = false
3511
3512   if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
3513   if len(argv) <= 0 {
3514     return false
3515   }
3516
3517   xargv := []string{}
3518   for ai := 0; ai < len(argv); ai++ {
3519     xargv = append(xargv,strsubst(gshCtx,argv[ai],false))
3520   }
3521   argv = xargv
3522   if false {
3523     for ai := 0; ai < len(argv); ai++ {
3524       fmt.Printf("[%d] %s [%d]\n",
3525           ai,argv[ai],len(argv[ai]),argv[ai])
3526     }
3527   }
3528   cmd := argv[0]
3529   if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)%v\n",len(argv),argv) }
3530   switch { // https://tour.golang.org/flowcontrol/11
3531     case cmd == "":
3532       gshCtx.xPwd([]string{}); // emtpy command
3533     case cmd == "x":
3534       gshCtx.CmdTrace = ! gshCtx.CmdTrace
3535     case cmd == "-xt":
3536       gshCtx.CmdTime = ! gshCtx.CmdTime
3537     case cmd == "-ot":
3538       gshCtx.sconnect(true, argv)
3539     case cmd == "-ou":
3540       gshCtx.sconnect(false, argv)
3541     case cmd == "-it":
3542       gshCtx.saccept(true , argv)
3543     case cmd == "-iu":
3544       gshCtx.saccept(false, argv)
3545     case cmd == "-i" || cmd == "<" || cmd == "-o" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
3546       gshCtx.redirect(argv)
3547     case cmd == "|":
3548       gshCtx.frompipe(argv)
3549     case cmd == "args":
3550       gshCtx.Args(argv)
3551     case cmd == "bg" || cmd == "-bg":
3552       rfin := gshCtx.inBackground(argv[1:])
3553       return rfin
3554     case cmd == "-bn":
3555       gshCtx.Basename(argv)
3556     case cmd == "call":
3557       _/_ = gshCtx.excommand(false,argv[1:])
3558     case cmd == "cd" || cmd == "chdir":
3559       gshCtx.xChdir(argv);
3560     case cmd == "-cksum":
3561       gshCtx.xFind(argv)
3562     case cmd == "-sum":
3563       gshCtx.xFind(argv)
3564     case cmd == "-sumtest":
3565       str := ""
3566       if 1 < len(argv) { str = argv[1] }
3567       crc := strCRC32(str,uint64(len(str)))
3568       fmt.Printf(stderr,"%v %v\n",crc,len(str))
3569     case cmd == "close":
3570       gshCtx.xClose(argv)
3571     case cmd == "gcp":
3572       gshCtx.FileCopy(argv)
3573     case cmd == "dec" || cmd == "decode":
3574       gshCtx.Dec(argv)
3575     case cmd == "#define":
3576     case cmd == "dic" || cmd == "d":
3577       xDic(argv)
3578     case cmd == "dump":
3579       gshCtx.Dump(argv)
3580     case cmd == "echo" || cmd == "e":
3581       echo(argv,true)
3582     case cmd == "enc" || cmd == "encode":
3583       gshCtx.Enc(argv)
3584     case cmd == "env":
3585       env(argv)
3586     case cmd == "eval":
3587       xEval(argv[1:],true)
3588     case cmd == "ev" || cmd == "events":
3589       dumpEvents(argv)
3590     case cmd == "exec":
3591       _/_ = gshCtx.excommand(true,argv[1:])
3592     // should not return here
3593     case cmd == "exit" || cmd == "quit":
3594     // write Result code EXIT to 3>
3595       return true
3596     case cmd == "fdls":

```

```

3597     // dump the attributes of fds (of other process)
3598     case cmd == "find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
3599         gshCtx.xFind(argv[1:])
3600     case cmd == "fu":
3601         gshCtx.xFind(argv[1:])
3602     case cmd == "fork":
3603         // mainly for a server
3604     case cmd == "gen":
3605         gshCtx.gen(argv)
3606     case cmd == "go":
3607         gshCtx.xGo(argv)
3608     case cmd == "grep":
3609         gshCtx.xFind(argv)
3610     case cmd == "dqe":
3611         gshCtx.Dqe(argv)
3612     case cmd == "genq":
3613         gshCtx.Erq(argv)
3614     case cmd == "gpop":
3615         gshCtx.Pop(argv)
3616     case cmd == "gpush":
3617         gshCtx.Push(argv)
3618     case cmd == "history" || cmd == "hi": // hi should be alias
3619         gshCtx.xHistory(argv)
3620     case cmd == "jobs":
3621         gshCtx.xJobs(argv)
3622     case cmd == "lsp" || cmd == "nlsp":
3623         gshCtx.SplitLine(argv)
3624     case cmd == "ls":
3625         gshCtx.xFind(argv)
3626     case cmd == "nop":
3627         // do nothing
3628     case cmd == "pipe":
3629         gshCtx.xOpen(argv)
3630     case cmd == "plug" || cmd == "plugin" || cmd == "pin":
3631         gshCtx.xPlugin(argv[1:])
3632     case cmd == "print" || cmd == "-pr":
3633         // output internal slice // also sprintf should be
3634         gshCtx.Println(argv)
3635     case cmd == "ps":
3636         gshCtx.xPs(argv)
3637     case cmd == "pstitle":
3638         // to be gsh.title
3639     case cmd == "rexecd" || cmd == "rexd":
3640         gshCtx.RexecServer(argv)
3641     case cmd == "rexec" || cmd == "rex":
3642         gshCtx.RexecClient(argv)
3643     case cmd == "repeat" || cmd == "rep": // repeat cond command
3644         gshCtx.repeat(argv)
3645     case cmd == "replay":
3646         gshCtx.XReplay(argv)
3647     case cmd == "scan":
3648         // scan input (or so in fscanf) to internal slice (like Files or map)
3649         gshCtx.Scanv(argv)
3650     case cmd == "set":
3651         // set name ...
3652     case cmd == "serv":
3653         gshCtx.httpServer(argv)
3654     case cmd == "shift":
3655         gshCtx.Shiftv(argv)
3656     case cmd == "sleep":
3657         gshCtx.Sleep(argv)
3658     case cmd == "sort":
3659         gshCtx.Sortv(argv)
3660
3661     case cmd == "j" || cmd == "join":
3662         gshCtx.Rjoin(argv)
3663     case cmd == "a" || cmd == "alpa":
3664         gshCtx.Rexec(argv)
3665     case cmd == "jcd" || cmd == "jchdir":
3666         gshCtx.Rchdir(argv)
3667     case cmd == "jget":
3668         gshCtx.Rget(argv)
3669     case cmd == "jls":
3670         gshCtx.Rls(argv)
3671     case cmd == "jput":
3672         gshCtx.Rput(argv)
3673     case cmd == "jpwd":
3674         gshCtx.Rpwd(argv)
3675
3676     case cmd == "time":
3677         fin = gshCtx.XTime(argv)
3678     case cmd == "ungets":
3679         if 1 < len(argv) {
3680             ungets(argv[1] + "\n")
3681         } else {
3682         }
3683     case cmd == "pwd":
3684         gshCtx.xPwd(argv);
3685     case cmd == "ver" || cmd == "-ver" || cmd == "version":
3686         gshCtx.showVersion(argv)
3687     case cmd == "where":
3688         // data file or so?
3689     case cmd == "which":
3690         which("PATH", argv);
3691     case cmd == "gj" && 1 < len(argv) && argv[1] == "listen":
3692         go gj_server(argv[1:]);
3693     case cmd == "gj" && 1 < len(argv) && argv[1] == "serve":
3694         go gj_server(argv[1:]);
3695     case cmd == "gj" && 1 < len(argv) && argv[1] == "join":
3696         go gj_client(argv[1:]);
3697     case cmd == "gj":
3698         jsend(argv);
3699     case cmd == "jsend":
3700         jsend(argv);
3701     default:
3702         if gshCtx.whichPlugin(cmd, []string{"-s"}) != nil {
3703             gshCtx.xPlugin(argv)
3704         } else {
3705             notfound,_ := gshCtx.excommand(false, argv)
3706             if notfound {
3707                 fmt.Printf("--E-- command not found (%v)\n", cmd)
3708             }
3709         }
3710     }
3711 }
3712 }
3713
3714 func (gsh*GshContext)gshell(gline string) (rfin bool) {
3715     argv := strings.Split(string(gline), " ")
3716     fin := gsh.gshell(argv)
3717     return fin
3718 }
3719 func (gsh*GshContext)tgshell(gline string)(xfin bool){
3720     start := time.Now()

```

```

3721     fin := gsh.gshelll(gline)
3722     end := time.Now()
3723     elps := end.Sub(start);
3724     if gsh.CmdTime {
3725         fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + "(%d.%09ds)\n",
3726             elps/1000000000,elps*1000000000)
3727     }
3728     return fin
3729 }
3730 func Ttyid() (int) {
3731     fi, err := os.Stdin.Stat()
3732     if err != nil {
3733         return 0;
3734     }
3735     //fmt.Printf("Stdin: %v Dev=%d\n",
3736     // fi.Mode(),fi.Mode()&os.ModeDevice)
3737     if (fi.Mode() & os.ModeDevice) != 0 {
3738         stat := syscall.Stat_t{};
3739         err := syscall.Fstat(0,&stat)
3740         if err != nil {
3741             //fmt.Printf("--I-- Stdin: (%v)\n",err)
3742         }else{
3743             //fmt.Printf("--I-- Stdin: rdev=%d %d\n",
3744             // stat.Rdev&0xFF,stat.Rdev);
3745             //fmt.Printf("--I-- Stdin: tty%d\n",stat.Rdev&0xFF);
3746             return int(stat.Rdev & 0xFF)
3747         }
3748     }
3749     return 0
3750 }
3751 func (gshCtx *GshContext) ttyfile() string {
3752     //fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
3753     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
3754         fmt.Sprintf("%02d",gshCtx.TerminalId)
3755         //strconv.Itoa(gshCtx.TerminalId)
3756     //fmt.Printf("--I-- ttyfile=%s\n",ttyfile)
3757     return ttyfile
3758 }
3759 func (gshCtx *GshContext) ttyline((*os.File){
3760     file, err := os.OpenFile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
3761     if err != nil {
3762         fmt.Printf("--F-- cannot open %s (%s)\n",gshCtx.ttyfile(),err)
3763         return file;
3764     }
3765     return file
3766 }
3767 func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string) {
3768     if( skipping ){
3769         reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
3770         line, _, _ := reader.ReadLine()
3771         return string(line)
3772     }else{
3773         if true {
3774             return xgetline(hix,prevline,gshCtx)
3775         }
3776     }
3777     else
3778     if( with_exgetline && gshCtx.GetLine != "" ){
3779         //var xhix int64 = int64(hix); // cast
3780         newenv := os.Environ()
3781         newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )
3782
3783         tty := gshCtx.ttyline()
3784         tty.WriteString(prevline)
3785         Pa := os.ProcAttr {
3786             "", // start dir
3787             newenv, //os.Environ(),
3788             []*os.File{os.Stdin,os.Stdout,os.Stderr,tty},
3789             nil,
3790         }
3791     //fmt.Printf("--I-- getline=%s // %s\n",gsh_getlinev[0],gshCtx.GetLine)
3792     proc, err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&Pa)
3793     if err != nil {
3794         fmt.Printf("--F-- getline process error (%v)\n",err)
3795         // for ; ; {
3796         return "exit (getline program failed)"
3797     }
3798     //stat, err := proc.Wait()
3799     proc.Wait()
3800     buff := make([]byte,LINESIZE)
3801     count, err := tty.Read(buff)
3802     //_, err = tty.Read(buff)
3803     //fmt.Printf("--D-- getline (%d)\n",count)
3804     if err != nil {
3805         if !(count == 0) { // && err.String() == "EOP" } {
3806             fmt.Printf("--E-- getline error (%s)\n",err)
3807         }
3808     }else{
3809         //fmt.Printf("--I-- getline OK \"%s\"\n",buff)
3810     }
3811     tty.Close()
3812     gline := string(buff[0:count])
3813     return gline
3814 }
3815 */
3816 {
3817     // if isatty {
3818         fmt.Printf("!%d",hix)
3819         fmt.Println(PROMPT)
3820     //}
3821     reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
3822     line, _, _ := reader.ReadLine()
3823     return string(line)
3824 }
3825 }
3826
3827 //== begin ===== getline =====
3828 /*
3829 * getline.c
3830 * 2020-0819 extracted from dog.c
3831 * getline.go
3832 * 2020-0822 ported to Go
3833 */
3834 */
3835 package main // getline main
3836 import (
3837     "fmt"      // <a href="https://golang.org/pkg/fmt/">fmt</a>
3838     "strings"   // <a href="https://golang.org/pkg/strings/">strings</a>
3839     "os"        // <a href="https://golang.org/pkg/os/">os</a>
3840     "syscall"   // <a href="https://golang.org/pkg/syscall/">syscall</a>
3841     //"bytes"    // <a href="https://golang.org/pkg/os/">os</a>
3842     //"os/exec"  // <a href="https://golang.org/pkg/os/">os</a>
3843 )
3844 */

```

```

3845 // C language compatibility functions
3846 var errno = 0
3847 var stdin *os.File = os.Stdin
3848 var stdout *os.File = os.Stdout
3849 var stderr *os.File = os.Stderr
3850 var EOF = -1
3851 var NULL = 0
3852 type FILE os.File
3853 type StrBuff []byte
3854 var NULL_FP *os.File = nil
3855 var NULLSP = 0
3856 //var LINESIZE = 1024
3857
3858 func system(cmdstr string)(int){
3859     PA := syscall.ProcAttr {
3860         "" // the starting directory
3861         os.Environ(),
3862         {uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd()},
3863         nil,
3864     }
3865     argv := strings.Split(cmdstr, " ")
3866     pid,err := syscall.ForkExec(argv[0],argv,&PA)
3867     if( err != nil ){
3868         fmt.Printf("--E-- syscall(%v) err(%v)\n",cmdstr,err)
3869     }
3870     syscall.Wait4(pid,nil,0,nil)
3871
3872 /*
3873 argv := strings.Split(cmdstr, " ")
3874 fmt.Fprintf(os.Stderr,"--I-- system(%v)\n",argv)
3875 //cmd := exec.Command(argv[0]...)
3876 cmd := exec.Command(argv[0],argv[1],argv[2])
3877 cmd.Stdin = strings.NewReader("output of system")
3878 var out bytes.Buffer
3879 cmd.Stdout = &out
3880 var serr bytes.Buffer
3881 cmd.Stderr = &serr
3882 err := cmd.Run()
3883 if err != nil {
3884     fmt.Fprintf(os.Stderr,"--E-- system(%v)err(%v)\n",argv,err)
3885     fmt.Println("ERR:",serr.String())
3886 }else{
3887     fmt.Printf("%s",out.String())
3888 }
3889 */
3890
3891 return 0
3892 }
3893 func atoi(str string)(ret int){
3894     ret,err := fmt.Sscanf(str,"%d",ret)
3895     if err == nil {
3896         return ret
3897     }else{
3898         // should set errno
3899         return 0
3900     }
3901 }
3902 func getenv(name string)(string){
3903     val,got := os.LookupEnv(name)
3904     if got {
3905         return val
3906     }else{
3907         return "?"
3908     }
3909 }
3910 func strcpy(dst StrBuff, src string){
3911     var i int
3912     srcb := []byte(src)
3913     for i = 0; i < len(src) && srcb[i] != 0; i++ {
3914         dst[i] = srcb[i]
3915     }
3916     dst[i] = 0
3917 }
3918 func xstrcpy(dst StrBuff, src StrBuff){
3919     dst = src
3920 }
3921 func strcat(dst StrBuff, src StrBuff){
3922     dst = append(dst,src...)
3923 }
3924 func strdup(str StrBuff)(string){
3925     return string(str[0:strlen(str)])
3926 }
3927 func strlen(str string)(int){
3928     return len(str)
3929 }
3930 func strlen(str StrBuff)(int){
3931     var i int
3932     for i = 0; i < len(str) && str[i] != 0; i++ {
3933     }
3934     return i
3935 }
3936 func sizeof(data StrBuff)(int){
3937     return len(data)
3938 }
3939 func isatty(fd int)(ret int){
3940     return 1
3941 }
3942
3943 func fopen(file string,mode string)(fp*os.File){
3944     if mode == "r" {
3945         fp,err := os.Open(file)
3946         if( err != nil ){
3947             fmt.Printf("--E-- fopen(%s,%s)=(%v)\n",file,mode,err)
3948             return NULL_FP;
3949         }
3950         return fp;
3951     }else{
3952         fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
3953         if( err != nil ){
3954             return NULL_FP;
3955         }
3956         return fp;
3957     }
3958 }
3959 func fclose(fp*os.File){
3960     fp.Close()
3961 }
3962 func fflush(fp *os.File)(int){
3963     return 0
3964 }
3965 func fgetc(fp*os.File)(int){
3966     var buf [1]byte
3967     _err := fp.Read(buf[0:1])
3968     if( _err != nil ){

```

```

3969     return EOF;
3970 }else{
3971     return int(buf[0])
3972 }
3973 }
3974 func sfgets(str*string, size int, fp*os.File)(int){
3975     buf := make(StrBuff,size)
3976     var ch int
3977     var i int
3978     for i = 0; i < len(buf)-1; i++ {
3979         ch = fgetc(fp)
3980         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
3981         if( ch == EOF ){
3982             break;
3983         }
3984         buf[i] = byte(ch);
3985         if( ch == '\n' ){
3986             break;
3987         }
3988     }
3989     buf[i] = 0
3990     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
3991     return i
3992 }
3993 func fgets(buf StrBuff, size int, fp*os.File)(int){
3994     var ch int
3995     var i int
3996     for i = 0; i < len(buf)-1; i++ {
3997         ch = fgetc(fp)
3998         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
3999         if( ch == EOF ){
4000             break;
4001         }
4002         buf[i] = byte(ch);
4003         if( ch == '\n' ){
4004             break;
4005         }
4006     }
4007     buf[i] = 0
4008     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4009     return i
4010 }
4011 func fputc(ch int , fp*os.File)(int){
4012     var buf [1]byte
4013     buf[0] = byte(ch)
4014     fp.Write(buf[0:1])
4015     return 0
4016 }
4017 func fputs(buf StrBuff, fp*os.File)(int){
4018     fp.Write(buf)
4019     return 0
4020 }
4021 func xputss(str string, fp*os.File)(int){
4022     return fputs([]byte(str),fp)
4023 }
4024 func sscanf(str StrBuff,fmts string, params ...interface{})(int){
4025     fmt.Sscanf(string(str[0:strlen(str)]),fmts,params...)
4026     return 0
4027 }
4028 func fprintf(fp*os.File,fmts string, params ...interface{})(int){
4029     fmt.Fprintf(fp,fmts,params...)
4030     return 0
4031 }
4032
4033 // <a name="IME">Command Line IME</a>
4034 //----- MyIME
4035 var MyIMEVER = "MyIME/0.0.2";
4036 type RomKana struct {
4037     dic string // dictionary ID
4038     pat string // input pattern
4039     out string // output pattern
4040     hit int64 // count of hit and used
4041 }
4042 var dicents = 0
4043 var romkana [1024]RomKana
4044 var Romkan []RomKana
4045
4046 func isinDic(str string)(int){
4047     for i,v := range Romkan {
4048         if v.pat == str {
4049             return i
4050         }
4051     }
4052     return -1
4053 }
4054 const (
4055     DIC_COM_LOAD = "im"
4056     DIC_COM_DUMP = "s"
4057     DIC_COM_LIST = "ls"
4058     DIC_COM_ENA = "en"
4059     DIC_COM_DIS = "di"
4060 )
4061 func helpDic(argv []string{
4062     out := stderr
4063     cmd := ""
4064     if 0 < len(argv) { cmd = argv[0] }
4065     fprintf(out,"-- %v Usage\n",cmd)
4066     fprintf(out,... Commands\n")
4067     fprintf(out,... %v %v [dicName] [dicURL] -- Import dictionary\n",cmd,DIC_COM_LOAD)
4068     fprintf(out,... %v %v [pattern] -- Search in dictionary\n",cmd,DIC_COM_DUMP)
4069     fprintf(out,... %v %v [dicName] -- List dictionaries\n",cmd,DIC_COM_LIST)
4070     fprintf(out,... %v %v [dicName] -- Disable dictionaries\n",cmd,DIC_COM_DIS)
4071     fprintf(out,... %v %v [dicName] -- Enable dictionaries\n",cmd,DIC_COM_ENA)
4072     fprintf(out,... Keys ... %v\n",ESC can be used for '\\\'')
4073     fprintf(out,... \\\c -- Reverse the case of the last character\n",)
4074     fprintf(out,... \\\i -- Replace input with translated text\n",)
4075     fprintf(out,... \\\j -- On/Off translation mode\n",)
4076     fprintf(out,... \\\l -- Force Lower Case\n",)
4077     fprintf(out,... \\\u -- Force Upper Case (software CapsLock)\n",)
4078     fprintf(out,... \\\v -- Show translation actions\n",)
4079     fprintf(out,... \\\x -- Replace the last input character with it Hexa-Decimal\n",)
4080 }
4081 func xDic(argv[]string{
4082     if len(argv) <= 1 {
4083         helpDic(argv)
4084         return
4085     }
4086     argv = argv[1:]
4087     var debug = false
4088     var info = false
4089     var silent = false
4090     var dump = false
4091     var builtin = false
4092     cmd := argv[0]

```

```

4093     argv = argv[1:]
4094     opt := ""
4095     arg := ""
4096
4097     if 0 < len(argv) {
4098         arg1 := argv[0]
4099         if arg1[0] == '-' {
4100             switch arg1 {
4101                 default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
4102                     return
4103                 case "-b": builtin = true
4104                 case "-d": debug = true
4105                 case "-s": silent = true
4106                 case "-v": info = true
4107             }
4108             opt = arg1
4109             argv = argv[1:]
4110         }
4111     }
4112
4113     dicName := ""
4114     dicURL := ""
4115     if 0 < len(argv) {
4116         arg = argv[0]
4117         dicName = arg
4118         argv = argv[1:]
4119     }
4120     if 0 < len(argv) {
4121         dicURL = argv[0]
4122         argv = argv[1:]
4123     }
4124     if false {
4125         fprintf(stderr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
4126     }
4127     if cmd == DIC_COM_LOAD {
4128         //dicType := ""
4129         dicBody := ""
4130         if !builtin && dicName != "" && dicURL == "" {
4131             f,err := os.Open(dicName)
4132             if err == nil {
4133                 dicURL = dicName
4134             }else{
4135                 f,err = os.Open(dicName+".html")
4136                 if err == nil {
4137                     dicURL = dicName+".html"
4138                 }else{
4139                     f,err = os.Open("gshdic-"+dicName+".html")
4140                     if err == nil {
4141                         dicURL = "gshdic-"+dicName+".html"
4142                     }
4143                 }
4144             }
4145             if err == nil {
4146                 var buf = make([]byte,128*1024)
4147                 count,err := f.Read(buf)
4148                 f.Close()
4149                 if info {
4150                     fprintf(stderr,"--Id-- ReadDic(%v,%v)\n",count,err)
4151                 }
4152                 dicBody = string(buf[0:count])
4153             }
4154         }
4155         if dicBody == "" {
4156             switch arg {
4157                 default:
4158                     dicName = "WorldDic"
4159                     dicURL = WorldDic
4160                     if info {
4161                         fprintf(stderr,"--Id-- default dictionary \"%v\"\n",
4162                             dicName);
4163                     }
4164                 case "wnn":
4165                     dicName = "WnnDic"
4166                     dicURL = WnnDic
4167                 case "sumomo":
4168                     dicName = "SumomoDic"
4169                     dicURL = Sumomodic
4170                 case "sijimi":
4171                     dicName = "SijimiDic"
4172                     dicURL = SijimiDic
4173                 case "jkl":
4174                     dicName = "JKLJaDic"
4175                     dicURL = JA_JKLDic
4176             }
4177             if debug {
4178                 fprintf(stderr,"--Id-- %v URL=%v\n",dicName,dicURL);
4179             }
4180             dicv := strings.Split(dicURL,",")
4181             if debug {
4182                 fprintf(stderr,"--Id-- %v encoded data...\n",dicName)
4183                 fprintf(stderr,"Type: %v\n",dicv[0])
4184                 fprintf(stderr,"Body: %v\n",dicv[1])
4185                 fprintf(stderr,"%\n")
4186             }
4187             body,_ := base64.StdEncoding.DecodeString(dicv[1])
4188             dicBody = string(body)
4189         }
4190         if info {
4191             fmt.Printf("--Id-- %v %v\n",dicName,dicURL)
4192             fmt.Printf("%s\n",dicBody)
4193         }
4194         if debug {
4195             fprintf(stderr,"--Id-- dicName %v text...\n",dicName)
4196             fprintf(stderr,"%v\n",string(dicBody))
4197         }
4198         envv := strings.Split(dicBody,"\n");
4199         if info {
4200             fprintf(stderr,"--Id-- %v scan...\n",dicName);
4201         }
4202         var added int = 0
4203         var dup int = 0
4204         for i,v := range envv {
4205             var pat string
4206             var out string
4207             fmt.Sscanf(v,"%s %s",&pat,&out)
4208             if len(pat) <= 0 {
4209                 }else{
4210                     if 0 <= isinDic(pat) {
4211                         dup += 1
4212                         continue
4213                     }
4214                     romkana[dicents] = RomKana{dicName,pat,out,0}
4215                     dicents += 1
4216                     added += 1
4217                 }
4218             }
4219         }
4220     }
4221     if added > 0 {
4222         if info {
4223             fmt.Printf("--Id-- %v added %v\n",dicName,added)
4224         }
4225     }
4226 }

```

```

4217     Romkan = append(Romkan,RomKana{dicName,pat,out,0})
4218     if debug {
4219         fmt.Printf("[%3v]:[%2v]%-8v [%2v]%v\n",
4220             i,len(pat),pat,len(out),out)
4221     }
4222   }
4223   if !silent {
4224     url := dicURL
4225     if strBegins(url,"data:") {
4226       url = "builtin"
4227     }
4228     fprintf(stderr,"--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4229             dicName,added,dup,len(Romkan),url);
4230   }
4231 // should sort by pattern length for concrete match, for performance
4232   if debug {
4233     arg = "" // search pattern
4234     dump = true
4235   }
4236 }
4237 }
4238 if cmd == DIC_COM_DUMP || dump {
4239   fprintf(stderr,"--Id-- %v dump... %v entries:\n",dicName,len(Romkan));
4240   var match = 0
4241   for i := 0; i < len(Romkan); i++ {
4242     dic := Romkan[i].dic
4243     pat := Romkan[i].pat
4244     out := Romkan[i].out
4245     if arg == "" || 0 <= strings.Index(pat,arg) | 0 <= strings.Index(out,arg) {
4246       fmt.Printf("\\\\%v\\t%v [%2v]%-8v [%2v]%v\n",
4247           i,dic,len(pat),pat,len(out),out)
4248       match += 1
4249     }
4250   }
4251   fprintf(stderr,"--Id-- %v matched %v / %v entries:\n",arg,match,len(Romkan));
4252 }
4253 }
4254 func loadDefaultDic(dic int){
4255   if( 0 < len(Romkan) ) {
4256     return
4257   }
4258   //fprintf(stderr,"r\n")
4259   xDic([]string{"dic",DIC_COM_LOAD});
4260
4261   var info = false
4262   if info {
4263     fprintf(stderr,"--Id-- Conguratuations!! WorldDic is now activated.\r\n")
4264     fprintf(stderr,"--Id-- enter \"dic\" command for help.\r\n")
4265   }
4266 }
4267 func readDic()(int){
4268 /*
4269   var rk *os.File;
4270   var dic = "MyIME-dic.txt";
4271   //rk = fopen("romkana.txt","r");
4272   //rk = fopen("JK-JA-morse-dic.txt","r");
4273   rk = fopen(dic,"r");
4274   if( rk == NULL_PP ){
4275     if( true ){
4276       fprintf(stderr,"--%s-- Could not load %s\n",MyIMEVER,dic);
4277     }
4278     return -1;
4279   }
4280   if( true ){
4281     var di int;
4282     var line = make(StrBuff,1024);
4283     var pat string
4284     var out string
4285     for di = 0; di < 1024; di++ {
4286       if( fgets(line,sizeof(line),rk) == NULLSP ){
4287         break;
4288       }
4289       fmt.Sscanf(string(line[0:strlen(line)]),"s s",&pat,&out);
4290       //sscanf(line,"%s %[^\r\n]",&pat,&out);
4291       romkana[di].pat = pat;
4292       romkana[di].out = out;
4293       //fprintf(stderr,"--Dd- %-10s %s\n",pat,out)
4294     }
4295     dicents += di
4296     if( false ){
4297       fprintf(stderr,"--%s-- loaded romkana.txt [%d]\n",MyIMEVER,di);
4298       for di = 0; di < dicents; di++ {
4299         fprintf(stderr,
4300             "%s %s\n",romkana[di].pat,romkana[di].out);
4301       }
4302     }
4303   }
4304   fclose(rk);
4305
4306 //romkana[dicents].pat = "//ddump"
4307 //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
4308 */
4309   return 0;
4310 }
4311 func matchlen(stri string, pati string)(int){
4312   if strBegins(stri,pati) {
4313     return len(pati)
4314   }else{
4315     return 0
4316   }
4317 }
4318 func convs(src string)(string){
4319   var si int;
4320   var si_ = len(src);
4321   var di int;
4322   var mi int;
4323   var dstb []byte
4324
4325   for si = 0; si < sx; { // search max. match from the position
4326     if strBegins(src[si:], "%x/") {
4327       // %x/integer // s/a/b/
4328       ix := strings.Index(src[si+3:], "/")
4329       if 0 < ix {
4330         var iv int = 0
4331         //fmt.Sscanf(src[si+3:si+3+ix],"%d",&iv)
4332         fmt.Sscanf(src[si+3:si+3+ix],"%v",&iv)
4333         sval := fmt.Sprintf("%x",iv)
4334         bval := []byte(sval)
4335         dstb = append(dstb,bval...)
4336         si = si+3+ix+1
4337         continue
4338       }
4339     }
4340     if strBegins(src[si:], "%d/") {

```

```

4341     // %d/integer/ // s/a/b/
4342     ix := strings.Index(src[si+3:],"/")
4343     if 0 < ix {
4344         var iv int = 0
4345         fmt.Sscanf(src[si+3:si+3+ix],"%v",&iv)
4346         sval := fmt.Sprintf("%d",iv)
4347         bval := []byte(sval)
4348         dstb = append(dstb,bval...)
4349         si = si+3+ix+1
4350         continue
4351     }
4352     if strBegins(src[si:], "st") {
4353         now := time.Now()
4354         if true {
4355             date := now.Format(time.Stamp)
4356             dstb = append(dstb,[]byte(date)...)}
4357             si = si+3
4358         }
4359         continue
4360     }
4361     var maxlen int = 0;
4362     var len int;
4363     mi = -1;
4364     for di = 0; di < dicents; di++ {
4365         len = matchlen(src[si:],romkana[di].pat);
4366         if( maxlen < len ){
4367             maxlen = len;
4368             mi = di;
4369         }
4370     }
4371     if( 0 < maxlen ){
4372         out := romkana[mi].out;
4373         dstb = append(dstb,[]byte(out)...);
4374         si += maxlen;
4375     }else{
4376         dstb = append(dstb,src[si:])
4377         si += 1;
4378     }
4379 }
4380 }
4381 return string(dstb)
4382 }
4383 func trans(src string)(int){
4384     dst := convs(src);
4385     xifputss(dst,stderr);
4386     return 0;
4387 }
4388
4389 //----- LINEEDIT
4390 // "?" at the top of the line means searching history
4391
4392 // should be compatilbe with Telnet
4393 const (
4394     EV_MODE      = 255
4395     EV_IDLE      = 254
4396     EV_TIMEOUT   = 253
4397
4398     GO_UP        = 252    // k
4399     GO_DOWN      = 251    // j
4400     GO_RIGHT     = 250    // l
4401     GO_LEFT      = 249    // h
4402     DEL_RIGHT    = 248    // x
4403     GO_TOPL      = 'A'-0x40 // 0
4404     GO_ENDL      = 'E'-0x40 // $ 
4405
4406     GO_TOPW      = 239    // b
4407     GO_ENDW      = 238    // e
4408     GO_NEXTW     = 237    // w
4409
4410     GO_FORWCH    = 229    // f
4411     GO_PAIRCH    = 228    // %
4412
4413     GO_DEL       = 219    // d
4414
4415     HI_SRCH_FW   = 209    // /
4416     HI_SRCH_BK   = 208    // ?
4417     HI_SRCH_RFW  = 207    // n
4418     HI_SRCH_RBZ  = 206    // N
4419 )
4420
4421 // should return number of octets ready to be read immediately
4422 //fprintf(stderr,"%n--Select(%v %v)\n",err,r.Bits[0])
4423
4424
4425 var EventRecvFd = -1 // file descriptor
4426 var EventSendFd = -1
4427 const EventFdOffset = 1000000
4428 const NormalFdOffset = 100
4429
4430 func putEvent(event int, evarg int){
4431     if true {
4432         if EventRecvFd < 0 {
4433             var pv = []int{-1,-1}
4434             syscall.Pipe(pv)
4435             EventRecvFd = pv[0]
4436             EventSendFd = pv[1]
4437             //fmt.Printf("--De-- EventPipe created[%v,%v]\n",EventRecvFd,EventSendFd)
4438     }
4439     }else{
4440         if EventRecvFd < 0 {
4441             // the document differs from this spec
4442             // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
4443             sv,err := syscall.Socketpair(syscall.AF_UNIX,syscall.SOCK_STREAM,0)
4444             EventRecvFd = sv[0]
4445             EventSendFd = sv[1]
4446             if err != nil {
4447                 fmt.Printf("--De-- EventSock created[%v,%v](%v)\n",
4448                     EventRecvFd,EventSendFd,err)
4449             }
4450         }
4451     }
4452     var buf = []byte{ byte(event) }
4453     n,err := syscall.Write(EventSendFd,buf)
4454     if err != nil {
4455         fmt.Printf("--De-- putEvent[%v](%3v)(%v)\n",EventSendFd,event,n,err)
4456     }
4457 }
4458 func ungets(str string){
4459     for _,ch := range str {
4460         putEvent(int(ch),0)
4461     }
4462 }
4463 func (gsh*GshContext)xReplay(argv[]string){
4464     hix := 0

```

```

4465     tempo := 1.0
4466     xtempo := 1.0
4467     repeat := 1
4468
4469     for _,a := range argv { // tempo
4470         if strBegins(a,"x") {
4471             fmt.Sscanf(a[1:], "%f", &xtempo)
4472             tempo = 1 / xtempo
4473             //fprintf(stderr, "--Dr-- tempo=%v\n", a[2:], tempo);
4474         }else
4475             if strBegins(a,"r") { // repeat
4476                 fmt.Sscanf(a[1:], "%v", &repeat)
4477             }else
4478                 if strBegins(a,"!") {
4479                     fmt.Sscanf(a[1:], "%d", &hix)
4480                 }else{
4481                     fmt.Sscanf(a, "%d", &hix)
4482                 }
4483             }
4484         if hix == 0 || len(argv) <= 1 {
4485             hix = len(gsh.CommandHistory)-1
4486         }
4487         fmt.Printf("--Ir-- Replay(!v x%v r%v)\n",hix,xtempo,repeat)
4488         //dumpEvents(hix)
4489         //gsh.xScanReplay(hix,false,repeat,tempo,argv)
4490         go gsh.xScanReplay(hix,true,repeat,tempo,argv)
4491     }
4492
4493 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4494 // 2020-0827 GShell-0.2.3
4495 */
4496 func FpollInl(fp *os.File,usec int)(uintptr){
4497     nfd := 1
4498
4499     rdv := syscall.FdSet {}
4500     fd1 := fp.Fd()
4501     bank1 := fd1/32
4502     mask1 := int32(1 << fd1)
4503     rdv.Bits[bank1] = mask1
4504
4505     fd2 := -1
4506     bank2 := -1
4507     var mask2 int32 = 0
4508
4509     if 0 <= EventRecvFd {
4510         fd2 = EventRecvFd
4511         nfd = fd2 + 1
4512         bank2 = fd2/32
4513         mask2 = int32(1 << fd2)
4514         rdv.Bits[bank2] |= mask2
4515         //fmt.Printf("--De-- EventPoll mask added [%d][%v][%v]\n",fd2,bank2,mask2)
4516     }
4517
4518     tout := syscall.NsecToTimeval(int64(usec*1000))
4519     //n,err := syscall.Select(nfd,&rdv,nil,nil,&tout) // spec. mismatch
4520     err := syscall.Select(nfd,&rdv,nil,nil,&tout)
4521     if err != nil {
4522         //fmt.Printf("--De-- select() err(%v)\n",err)
4523     }
4524     if err == nil {
4525         if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4526             if false {
4527                 fmt.Printf("--De-- got Event\n")
4528             }
4529             return uintptr(EventFdOffset + fd2)
4530         }else
4531             if (rdv.Bits[bank1] & mask1) != 0 {
4532                 return uintptr(NormalFdOffset + fd1)
4533             }else{
4534                 return 1
4535             }
4536         }else{
4537             return 0
4538         }
4539     }
4540 */
4541 func fgetcTimeout1(fp *os.File,usec int)(int){
4542     READ1:
4543     //readyFd := FpollInl(fp,usec)
4544     readyFd := CFpollInl(fp,usec)
4545     if readyFd < 100 {
4546         return EV_TIMEOUT
4547     }
4548
4549     var buf [1]byte
4550
4551     if EventFdOffset <= readyFd {
4552         fd := int(readyFd-EventFdOffset)
4553         _,err := syscall.Read(fd,buf[0:1])
4554         if( err != nil ){
4555             return EOF;
4556         }else{
4557             if buf[0] == EV_MODE {
4558                 recvEvent(fd)
4559                 goto READ1
4560             }
4561             return int(buf[0])
4562         }
4563     }
4564
4565     _,err := fp.Read(buf[0:1])
4566     if( err != nil ){
4567         return EOF;
4568     }else{
4569         return int(buf[0])
4570     }
4571 }
4572
4573 func visibleChar(ch int)(string){
4574     switch {
4575         case '!' <= ch && ch <= '~':
4576             return string(ch)
4577     }
4578     switch ch {
4579         case ' ': return "\\s"
4580         case '\n': return "\\n"
4581         case '\r': return "\\r"
4582         case '\t': return "\\t"
4583     }
4584     switch ch {
4585         case 0x00: return "NUL"
4586         case 0x07: return "BEL"
4587         case 0x08: return "BS"
4588         case 0x0E: return "SO"
4589     }

```

```

4589     case 0x0F: return "SI"
4590     case 0x1B: return "ESC"
4591     case 0x7F: return "DEL"
4592   }
4593   switch ch {
4594     case EV_IDLE: return fmt.Sprintf("IDLE")
4595     case EV_MODE: return fmt.Sprintf("MODE")
4596   }
4597   return fmt.Sprintf("%c",ch)
4598 }
4599 func recvEvent(fd int){
4600   var buf = make([]byte,1)
4601   _,_ = syscall.Read(fd,buf[0:1])
4602   if( buf[0] != 0 ){
4603     romkanmode = true
4604   }else{
4605     romkanmode = false
4606   }
4607 }
4608 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
4609   var Start time.Time
4610   var events = []Event{
4611     for e := range Events {
4612       if hix == 0 || e.CmdIndex == hix {
4613         events = append(events,e)
4614       }
4615     }
4616   elen := len(events)
4617   if 0 < elen {
4618     if events[elen-1].event == EV_IDLE {
4619       events = events[0:elen-1]
4620     }
4621   }
4622   for r := 0; r < repeat; r++ {
4623     for i,e := range events {
4624       nano := e.when.Nanosecond()
4625       micro := nano / 1000
4626       if Start.Second() == 0 {
4627         Start = time.Now()
4628       }
4629       diff := time.Now().Sub(Start)
4630       if replay {
4631         if e.event != EV_IDLE {
4632           putEvent(e.event,0)
4633           if e.event == EV_MODE { // event with arg
4634             putEvent(int(e.evarg),0)
4635           }
4636         }
4637       }else{
4638         fmt.Printf("%7.3fms %3v !%3v [%v.%06d] %3v %02X %-4v %10.3fms\n",
4639           float64(diff)/1000000.0,
4640           i,
4641           e.CmdIndex,
4642           e.when.Format(time.Stamp),micro,
4643           e.event,e.event.visibleChar(e.event),
4644           float64(e.evarg)/1000000.0)
4645       }
4646       if e.event == EV_IDLE {
4647         d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
4648         //nsleep(time.Duration(e.evarg))
4649         nsleep(d)
4650       }
4651     }
4652   }
4653 }
4654 func dumpEvents(argv[]string){
4655   hix := 0
4656   if 1 < len(argv) {
4657     fmt.Sscanf(argv[1],"%d",&hix)
4658   }
4659   for i,e := range Events {
4660     nano := e.when.Nanosecond()
4661     micro := nano / 1000
4662     //if e.event != EV_TIMEOUT {
4663     if hix == 0 || e.CmdIndex == hix {
4664       fmt.Printf("%-3v !%-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",i,
4665           e.CmdIndex,
4666           e.when.Format(time.Stamp),micro,
4667           e.event,e.event.visibleChar(e.event),float64(e.evarg)/1000000.0)
4668     }
4669   //}
4670 }
4671 }
4672 func fgetcTimeout(fp *os.File,usec int)(int){
4673   ch := fgetcTimeout(fp,usec)
4674   if ch != EV_TIMEOUT {
4675     now := time.Now()
4676     if 0 < len(Events) {
4677       last := Events[len(Events)-1]
4678       dura := int64(now.Sub(last.when))
4679       Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
4680     }
4681     Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
4682   }
4683   return ch
4684 }
4685
4686 var AtConsoleLineTop = true
4687 var TtyMaxCol = 72 // to be obtained by ioctl?
4688 var EscTimeout = (100*1000)
4689 var (
4690   MODE_VicMode    bool    // vi compatible command mode
4691   MODE_ShowMode   bool    // shown translation mode, the mode to be retained
4692   romkanmode      bool    // shown translation mode, the mode to be retained
4693   MODE_Recursive   bool    // recursive translation
4694   MODE_CapsLock   bool    // software CapsLock
4695   MODE_LowerLock  bool    // force lower-case character lock
4696   MODE_ViInsert   int     // visible insert mode, should be like "I" icon in X Window
4697   MODE_ViTrace    bool    // output newline before translation
4698 )
4699 type IInput struct {
4700   lno  int
4701   lastlno int
4702   pch  []int // input queue
4703   prompt string
4704   line  string
4705   right string
4706   inJMode  bool
4707   pinJMode bool
4708   waitingMeta string // waiting meta character
4709   LastCmd string
4710 }
4711 func (iin*IInput)Getc(timeoutUs int)(int){
4712   ch1 := EOF

```

```

4713     ch2 := EOF
4714     ch3 := EOF
4715     if( 0 < len(iin.pch) ){ // deQ
4716         ch1 = iin.pch[0]
4717         iin.pch = iin.pch[1:]
4718     }else{
4719         ch1 = fgetcTimeout(stdin,timeoutUs);
4720     }
4721     if( ch1 == 033 ){ // escape sequence
4722         ch2 = fgetcTimeout(stdin,EscTimeout);
4723         if( ch2 == EV_TIMEOUT ){
4724             }else{
4725                 ch3 = fgetcTimeout(stdin,EscTimeout);
4726                 if( ch3 == EV_TIMEOUT ){
4727                     iin.pch = append(iin.pch,ch2) // enQ
4728                 }else{
4729                     switch( ch2 ){
4730                         default:
4731                             iin.pch = append(iin.pch,ch2) // enQ
4732                             iin.pch = append(iin.pch,ch3) // enQ
4733                         case '[':
4734                             switch( ch3 ){
4735                                 case 'A': ch1 = GO_UP; // ^
4736                                 case 'B': ch1 = GO_DOWN; // v
4737                                 case 'C': ch1 = GO_RIGHT; // >
4738                                 case 'D': ch1 = GO_LEFT; // <
4739                                 case '3':
4740                                     ch4 := fgetcTimeout(stdin,EscTimeout);
4741                                     if( ch4 == '~' ){
4742                                         //fprintf(stderr,"x[%02X %02X %02X]\n",ch1,ch2,ch3,ch4);
4743                                         ch1 = DEL_RIGHT
4744                                     }
4745                                     case '\\':
4746                                         //ch4 := fgetcTimeout(stdin,EscTimeout);
4747                                         //fprintf(stderr,"y[%02X %02X %02X]\n",ch1,ch2,ch3,ch4);
4748                                         switch( ch3 ){
4749                                             case '~': ch1 = DEL_RIGHT
4750                                         }
4751                                     }
4752                                 }
4753                             }
4754                         }
4755                     return ch1
4756                 }
4757             }
4758         func (iin*IInput)clearline(){
4759             var i int
4760             fprintf(stderr,"\r");
4761             // should be ANSI ESC sequence
4762             for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
4763                 fputc(' ',os.Stderr);
4764             }
4765             fprintf(stderr,"\r");
4766         }
4767         func (iin*IInput)Redraw(){
4768             redraw(iin,iin.lno,iin.line,iin.right)
4769         }
4770         func redraw(iin *IInput,lno int,line string,right string){
4771             inMeta := false
4772             showMode := ""
4773             showMeta := "" // visible Meta mode on the cursor position
4774             showLino := fmt.Sprintf("%d",lno)
4775             InsertMark := "" // in visible insert mode
4776
4777             if MODE_VicMode {
4778             }else
4779             if 0 < len(iin.right) {
4780                 InsertMark = " "
4781             }
4782
4783             if( 0 < len(iin.waitingMeta) ){
4784                 inMeta = true
4785                 if iin.waitingMeta[0] != 033 {
4786                     showMeta = iin.waitingMeta
4787                 }
4788             }
4789             if( romkanmode ){
4790                 /romkanmark = "*";
4791             }else{
4792                 /romkanmark = "";
4793             }
4794             if MODE_ShowMode {
4795                 romkan := "--"
4796                 inmeta := "."
4797                 inveri := "."
4798                 if MODE_CapsLock {
4799                     inmeta = "A"
4800                 }
4801                 if MODE_LowerLock {
4802                     inmeta = "a"
4803                 }
4804                 if MODE_ViTrace {
4805                     inveri = "v"
4806                 }
4807                 if MODE_VicMode {
4808                     inveri = ":"
4809                 }
4810                 if romkanmode {
4811                     romkan = "\343\201\202"
4812                     if MODE_CapsLock {
4813                         inmeta = "R"
4814                     }else{
4815                         inmeta = "r"
4816                     }
4817                 if inMeta {
4818                     inmeta = "\\"
4819                 }
4820                 showMode = "["+romkan+inmeta+inveri+"]";
4821             }
4822             Pre := "\r" + showMode + showLino
4823             Output := ""
4824             Left := ""
4825             Right := ""
4826             if romkanmode {
4827                 Left = convs(line)
4828                 Right = InsertMark+convs(right)
4829             }else{
4830                 Left = line
4831                 Right = InsertMark+right
4832             }
4833             Output = Pre+Left
4834             if MODE_ViTrace {
4835                 Output += iin.LastCmd
4836             }

```

```

4837     }
4838     Output += showMeta+Right
4839     for len(Output) < TtyMaxCol { // to the max. position that may be dirty
4840         Output += " "
4841         // should be ANSI ESC sequence
4842         // not necessary just after newline
4843     }
4844     Output += Pre+Left+showMeta // to set the cursor to the current input position
4845     fprintf(stderr,"%s",output)
4846
4847     if MODE_ViTrace {
4848         if 0 < len(iin.LastCmd) {
4849             iin.LastCmd = ""
4850             fprintf(stderr,"%r\n")
4851         }
4852     }
4853     AtConsoleLineTop = false
4854 }
4855 // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
4856 func delHeadChar(str string)(rline string,head string){
4857     _,clen := utf8.DecodeRune([]byte(str))
4858     head = string(str[0:clen])
4859     return str[clen:],head
4860 }
4861 func delTailChar(str string)(rline string, last string){
4862     var i = 0
4863     var clen = 0
4864     for {
4865         ,siz := utf8.DecodeRune([]byte(str)[i:])
4866         if siz <= 0 { break }
4867         clen = siz
4868         i += siz
4869     }
4870     last = str[len(str)-clen:]
4871     return str[0:len(str)-clen],last
4872 }
4873
4874 // 3> for output and history
4875 // 4> for keylog?
4876 // <a name="getline">Command Line Editor</a>
4877 func xgetline(lno int, prevline string, gsh*GshContext)(string){
4878     var iin IInput
4879     iin.lastlno = lno
4880     iin.lno = lno
4881
4882     CmdIndex = len(gsh.CommandHistory)
4883     if( isatty(0) == 0 ){
4884         if( sfgets(&iin.line,INESIZE,stdin) == NULL ){
4885             iin.line = "exit\n";
4886         }else{
4887         }
4888         return iin.line
4889     }
4890     if( true ){
4891         //var pts string;
4892         //pts = ptsname(0);
4893         //pts = ttyname(0);
4894         //fprintf(stderr,"--pts[0] = %s\n",pts?pts:"?");
4895     }
4896     if( false ){
4897         fprintf(stderr,"! ");
4898         fflush(stderr);
4899         sfgets(&iin.line,INESIZE,stdin);
4900         return iin.line
4901     }
4902     system("/bin/stty -echo -icanon");
4903     xline := iin.xgetline(prevline,gsh)
4904     system("/bin/stty echo sane");
4905     return xline
4906 }
4907 func (iin*IInput)Translate(cmdch int){
4908     romkanemode = !romkanemode;
4909     if MODE_ViTrace {
4910         fprintf(stderr,"%v\r\n",string(cmdch));
4911     }else
4912     if( cmdch == 'J' ){
4913         fprintf(stderr,"J\r\n");
4914         iin.inJMode = true
4915     }
4916     iin.Redraw();
4917     loaddefaultdic(cmdch);
4918     iin.Redraw();
4919 }
4920 func (iin*IInput)Replace(cmdch int){
4921     iin.LastCmd = fmt.Sprintf("\\"%v",string(cmdch))
4922     iin.Redraw();
4923     loaddefaultdic(cmdch);
4924     dst := convs(in.line+iin.right);
4925     iin.line = dst
4926     iin.right = ""
4927     if( cmdch == 'I' ){
4928         fprintf(stderr,"I\r\n");
4929         iin.inJMode = true
4930     }
4931     iin.Redraw();
4932 }
4933 // aa 12 aia1
4934 func isAlpha(ch rune)(bool){
4935     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
4936         return true
4937     }
4938     return false
4939 }
4940 func isAlnum(ch rune)(bool){
4941     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'z' {
4942         return true
4943     }
4944     if '0' <= ch && ch <= '9' {
4945         return true
4946     }
4947     return false
4948 }
4949
4950 // 0.2.8 2020-0901 created
4951 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
4952 func (iin*IInput)GotoTOPW(){
4953     str := iin.line
4954     i := len(str)
4955     if i <= 0 {
4956         return
4957     }
4958     //i0 := i
4959     i -= 1
4960     lastSize := 0

```

```

4961 var lastRune rune
4962 var found = -1
4963 for 0 < i { // skip preamble spaces
4964     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4965     if !isAlnum(lastRune) { // character, type, or string to be searched
4966         i -= lastSize
4967         continue
4968     }
4969     break
4970 }
4971 for 0 < i {
4972     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4973     if lastSize <= 0 { continue } // not the character top
4974     if !isAlnum(lastRune) { // character, type, or string to be searched
4975         found = i
4976         break
4977     }
4978     i -= lastSize
4979 }
4980 if found < 0 && i == 0 {
4981     found = 0
4982 }
4983 if 0 <= found {
4984     if isAlnum(lastRune) { // or non-kana character
4985     }else{ // when positioning to the top o the word
4986         i += lastSize
4987     }
4988     iin.right = str[i:] + iin.right
4989     if 0 < i {
4990         iin.line = str[0:i]
4991     }else{
4992         iin.line = ""
4993     }
4994 }
4995 //fmt.Printf("\n(%d,%d,%d)[%s][%s]\n",i,found,iin.line,iin.right)
4996 //fmt.Printf("") // set debug messae at the end of line
4997 }
4998 // 0.2.0 2020-0901 created
4999 func (iin*IInput)GotoENDW(){
5000     str := iin.right
5001     if len(str) <= 0 {
5002         return
5003     }
5004     lastSize := 0
5005     var lastRune rune
5006     var lastW = 0
5007     i := 0
5008     inWord := false
5009
5010     lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
5011     if isAlnum(lastRune) {
5012         r,z := utf8.DecodeRuneInString(str[lastSize:])
5013         if 0 < z && isAlnum(r) {
5014             inWord = true
5015         }
5016     }
5017     for i < len(str) {
5018         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5019         if lastSize <= 0 { break } // broken data?
5020         if !isAlnum(lastRune) { // character, type, or string to be searched
5021             break
5022         }
5023         lastW = i // the last alnum if in alnum word
5024         i += lastSize
5025     }
5026     if inWord {
5027         goto DISP
5028     }
5029     for i < len(str) {
5030         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5031         if lastSize <= 0 { break } // broken data?
5032         if isAlnum(lastRune) { // character, type, or string to be searched
5033             break
5034         }
5035         i += lastSize
5036     }
5037     for i < len(str) {
5038         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5039         if lastSize <= 0 { break } // broken data?
5040         if !isAlnum(lastRune) { // character, type, or string to be searched
5041             break
5042         }
5043         lastW = i
5044         i += lastSize
5045     }
5046 DISP:
5047     if 0 < lastW {
5048         iin.line = iin.line + str[0:lastW]
5049         iin.right = str[lastW:]
5050     }
5051 //fmt.Printf("\n(%d)[%s]\n",i,iin.line,iin.right)
5052 //fmt.Printf("") // set debug messae at the end of line
5053 }
5054 // 0.2.8 2020-0901 created
5055 func (iin*IInput)GotoNEXTW(){
5056     str := iin.right
5057     if len(str) <= 0 {
5058         return
5059     }
5060     lastSize := 0
5061     var lastRune rune
5062     var found = -1
5063     i := 1
5064     for i < len(str) {
5065         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5066         if lastSize <= 0 { break } // broken data?
5067         if !isAlnum(lastRune) { // character, type, or string to be searched
5068             found = i
5069             break
5070         }
5071         i += lastSize
5072     }
5073     if 0 < found {
5074         if isAlnum(lastRune) { // or non-kana character
5075         }else{ // when positioning to the top o the word
5076             found += lastSize
5077         }
5078         iin.line = iin.line + str[0:found]
5079         if 0 < found {
5080             iin.right = str[found:]
5081         }else{
5082             iin.right = ""
5083         }
5084 }

```

```

5085     //fmt.Printf("\n%d[%s]\n",i,in.line,in.right)
5086     //fmt.Println("") // set debug messae at the end of line
5087 }
5088 // 0.2.8 2020-0902 created
5089 func (iin*IInput)GotoPAIRCH(){
5090     str := iin.right
5091     if len(str) <= 0 {
5092         return
5093     }
5094     lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
5095     if lastSize <= 0 {
5096         return
5097     }
5098     forw := false
5099     back := false
5100     pair := ""
5101     switch string(lastRune){
5102         case "(": pair = ")"; forw = true
5103         case ")": pair = "("; back = true
5104         case "[": pair = "]"; forw = true
5105         case "]": pair = "["; back = true
5106         case "{": pair = "}"; forw = true
5107         case "}": pair = "{"; back = true
5108         case "<": pair = ">"; forw = true
5109         case ">": pair = "<"; back = true
5110         case "<>": pair = "<>"; back = true
5111         case "'": pair = "\""; // context depednet, can be f" or back-double quote
5112         case '\"': pair = """; // context depednet, can be f' or back-quote
5113         // case Japanese Kakkos
5114     }
5115     if forw {
5116         iin.SearchForward(pair)
5117     }
5118     if back {
5119         iin.SearchBackward(pair)
5120 }
5121 // 0.2.8 2020-0902 created
5122 func (iin*IInput)SearchForward(pat string)(bool){
5123     right := iin.right
5124     found := -1
5125     i := 0
5126     if strBegins(right,pat) {
5127         z := utf8.DecodeRuneInString(right[i:])
5128         if 0 < z {
5129             i += z
5130         }
5131     }
5132     for i < len(right) {
5133         if strBegins(right[i:],pat) {
5134             found = i
5135             break
5136         }
5137         z := utf8.DecodeRuneInString(right[i:])
5138         if z <= 0 { break }
5139         i += z
5140     }
5141     if 0 <= found {
5142         iin.line = iin.line + right[0:found]
5143         iin.right = iin.right[found:]
5144         return true
5145     }else{
5146         return false
5147     }
5148 }
5149 // 0.2.8 2020-0902 created
5150 func (iin*IInput)SearchBackward(pat string)(bool){
5151     line := iin.line
5152     found := -1
5153     i := len(line)-1
5154     for i = i; 0 <= i; i-- {
5155         z := utf8.DecodeRuneInString(line[i:])
5156         if z <= 0 {
5157             continue
5158         }
5159         //fprintf(stderr,"-- %v %v\n",pat,line[i:])
5160         if strBegins(line[i:],pat) {
5161             found = i
5162             break
5163         }
5164     }
5165     //fprintf(stderr,"--%d\n",found)
5166     if 0 <= found {
5167         iin.right = line[found:] + iin.right
5168         iin.line = line[0:found]
5169         return true
5170     }else{
5171         return false
5172     }
5173 }
5174 // 0.2.8 2020-0902 created
5175 // search from top, end, or current position
5176 func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool,string){
5177     if forw {
5178         for _,v := range gsh.CommandHistory {
5179             if 0 <= strings.Index(v.CmdLine,pat) {
5180                 //fprintf(stderr,"\n-- De-- found !%v [%v]\n",i,pat,v.CmdLine)
5181                 return true,v.CmdLine
5182             }
5183         }
5184     }else{
5185         hlen := len(gsh.CommandHistory)
5186         for i := hlen-1; 0 < i; i-- {
5187             v := gsh.CommandHistory[i]
5188             if 0 <= strings.Index(v.CmdLine,pat) {
5189                 //fprintf(stderr,"\n-- De-- found !%v [%v]\n",i,pat,v.CmdLine)
5190                 return true,v.CmdLine
5191             }
5192         }
5193     }
5194     //fprintf(stderr,"-- not-found(%v)\n",pat)
5195     return false,"(Not Found in History)"
5196 }
5197 // 0.2.8 2020-0902 created
5198 func (iin*IInput)GotoFORSTR(pat string,gsh*GshContext){
5199     found := false
5200     if 0 < len(iin.right) {
5201         found = iin.SearchForward(pat)
5202     }
5203     if !found {
5204         found,line := gsh.SearchHistory(pat,true)
5205         if found {
5206             iin.line = line
5207             iin.right = ""
5208         }
5209 }

```

```

5209     }
5210 }
5211 func (iin*IInput)GotoBACKSTR(pat string, gsh*GshContext){
5212     found := false
5213     if 0 < len(iin.line) {
5214         found = iin.SearchBackward(pat)
5215     }
5216     if !found {
5217         found, line := gsh.SearchHistory(pat, false)
5218         if found {
5219             iin.line = line
5220             iin.right = ""
5221         }
5222     }
5223 }
5224 func (iin*IInput)getstringl(prompt string)(string){ // should be editable
5225     iin.clearline();
5226     fprintf(stderr,"r%v",prompt)
5227     str := ""
5228     for {
5229         ch := iin.Getc(10*1000*1000)
5230         if ch == '\n' || ch == '\r' {
5231             break
5232         }
5233         sch := string(ch)
5234         str += sch
5235         fprintf(stderr,"%s",sch)
5236     }
5237     return str
5238 }
5239 // search pattern must be an array and selectable with ^N/^P
5240 var SearchPat = ""
5241 var SearchForw = true
5242 var SearchForw = true
5243
5244 func (iin*IInput)xgetline1(prevline string, gsh*GshContext)(string){
5245     var ch int;
5246
5247     MODE_ShowMode = false
5248     MODE_VicMode = false
5249     iin.Redraw();
5250     first := true
5251
5252     for cix := 0; ; cix++ {
5253         iin.pinJmode = iin.inJmode
5254         iin.indMode = false
5255
5256         ch = iin.Getc(1000*1000)
5257
5258         if ch != EV_TIMEOUT && first {
5259             first = false
5260             mode := 0
5261             if romkanemode {
5262                 mode = 1
5263             }
5264             now := time.Now()
5265             Events = append(Events,Event{now,EV_MODE,int64(mode),CmdIndex})
5266         }
5267         if ch == 033 {
5268             MODE_ShowMode = true
5269             MODE_VicMode = !MODE_VicMode
5270             iin.Redraw();
5271             continue
5272         }
5273         if MODE_VicMode {
5274             switch ch {
5275                 case '0': ch = GO_TOPL
5276                 case 'S': ch = GO_ENDL
5277                 case 'B': ch = GO_TOPW
5278                 case 'E': ch = GO_ENDW
5279                 case 'W': ch = GO_NEXTW
5280                 case 'A': ch = GO_PAIRCH
5281
5282                 case 'j': ch = GO_DOWN
5283                 case 'k': ch = GO_UP
5284                 case 'h': ch = GO_LEFT
5285                 case 'l': ch = GO_RIGHT
5286                 case 'x': ch = DEL_RIGHT
5287                 case 'a': MODE_VicMode = !MODE_VicMode
5288                     ch = GO_RIGHT
5289                 case 'i': MODE_VicMode = !MODE_VicMode
5290                     iin.Redraw();
5291                     continue
5292                 case '-':
5293                     right,head := delHeadChar(iin.right)
5294                     if len([]byte(head)) == 1 {
5295                         ch = int(head[0])
5296                         if( 'A' <= ch && ch <= 'z' ){
5297                             ch = ch + 'A'-'a'
5298                         }else{
5299                             if( 'A' <= ch && ch <= 'Z' ){
5300                                 ch = ch + 'a'-'A'
5301                             }
5302                         iin.right = string(ch) + right
5303                     }
5304                     iin.Redraw();
5305                     continue
5306                 case 'f': // GO_FORWCH
5307                     iin.Redraw();
5308                     ch = iin.Getc(3*1000*1000)
5309                     if ch == EV_TIMEOUT {
5310                         iin.Redraw();
5311                         continue
5312                     }
5313                     SearchPat = string(ch)
5314                     SearchForw = true
5315                     iin.GotoFORWSTR(SearchPat,gsh)
5316                     iin.Redraw();
5317                     continue
5318                 case '/':
5319                     SearchPat = iin.getstringl("//") // should be editable
5320                     SearchForw = true
5321                     iin.GotoFORWSTR(SearchPat,gsh)
5322                     iin.Redraw();
5323                     continue
5324                 case '?':
5325                     SearchPat = iin.getstringl("?) // should be editable
5326                     SearchForw = false
5327                     iin.GotoBACKSTR(SearchPat,gsh)
5328                     iin.Redraw();
5329                     continue
5330                 case 'n':
5331                     if SearchForw {
5332                         iin.GotoFORWSTR(SearchPat,gsh)

```

```

5333         }else{
5334             iin.GotoBACKSTR(SearchPat,gsh)
5335         }
5336         iin.Redraw();
5337         continue
5338     case 'N':
5339         if !SearchForw {
5340             iin.GotoFORWSTR(SearchPat,gsh)
5341         }else{
5342             iin.GotoBACKSTR(SearchPat,gsh)
5343         }
5344         iin.Redraw();
5345         continue
5346     }
5347 }
5348 switch ch {
5349     case GO_TOPW:
5350         iin.GotoTOPW()
5351         iin.Redraw();
5352         continue
5353     case GO_ENDW:
5354         iin.GotoENDW()
5355         iin.Redraw();
5356         continue
5357     case GO_NEXTW:
5358         // to next space then
5359         iin.GotoNEXTW()
5360         iin.Redraw();
5361         continue
5362     case GO_PAIRCH:
5363         iin.GotoPAIRCH()
5364         iin.Redraw();
5365         continue
5366 }
5367
5368 //fprintf(stderr,"A[%02X]\n",ch);
5369 if( ch == '\\\' || ch == 033 ){
5370     MODE_ShowMode = true
5371     metach := ch
5372     iin.waitingMeta = string(ch)
5373     iin.Redraw();
5374     // set cursor //fprintf(stderr,"??\b\b\b")
5375     ch = fgetTimeout(stdin,2000*1000)
5376     // reset cursor
5377     iin.waitingMeta = ""
5378
5379 cmdch := ch
5380 if( ch == EV_TIMEOUT ){
5381     if metach == 033 {
5382         continue
5383     }
5384     ch = metach
5385 }
5386 /*
5387 if( ch == 'm' || ch == 'M' ){
5388     mch := fgetTimeout(stdin,1000*1000)
5389     if mch == 'r' {
5390         romkanmode = true
5391     }else{
5392         romkanmode = false
5393     }
5394     continue
5395 }
5396 */
5397 if( ch == 'k' || ch == 'K' ){
5398     MODE_Recursive = !MODE_Recursive
5399     iin.Translate(cmdch);
5400     continue
5401 }
5402 if( ch == 'j' || ch == 'J' ){
5403     iin.Translate(cmdch);
5404     continue
5405 }
5406 if( ch == 'i' || ch == 'I' ){
5407     iin.Replace(cmdch);
5408     continue
5409 }
5410 if( ch == 'l' || ch == 'L' ){
5411     MODE_LowerLock = !MODE_LowerLock
5412     MODE_CapsLock = false
5413     if MODE_ViTrace {
5414         fprintf(stderr,"%v\r\n",string(cmdch));
5415     }
5416     iin.Redraw();
5417     continue
5418 }
5419 if( ch == 'u' || ch == 'U' ){
5420     MODE_CapsLock = !MODE_CapsLock
5421     MODE_LowerLock = false
5422     if MODE_ViTrace {
5423         fprintf(stderr,"%v\r\n",string(cmdch));
5424     }
5425     iin.Redraw();
5426     continue
5427 }
5428 if( ch == 'v' || ch == 'V' ){
5429     MODE_ViTrace = !MODE_ViTrace
5430     if MODE_ViTrace {
5431         fprintf(stderr,"%v\r\n",string(cmdch));
5432     }
5433     iin.Redraw();
5434     continue
5435 }
5436 if( ch == 'c' || ch == 'C' ){
5437     if 0 < len(iin.line) {
5438         xline,tail := delTailChar(iin.line)
5439         if len([jbyte(tail)]) == 1 {
5440             ch = int(tail[0])
5441             if( 'a' <= ch && ch <= 'z' ){
5442                 ch = ch + 'A'-'a'
5443             }else{
5444                 if( 'A' <= ch && ch <= 'Z' ){
5445                     ch = ch + 'a'-'A'
5446                 }
5447                 iin.line = xline + string(ch)
5448             }
5449         }
5450         if MODE_ViTrace {
5451             fprintf(stderr,"%v\r\n",string(cmdch));
5452         }
5453         iin.Redraw();
5454         continue
5455     }
5456     iin.pch = append(iin.pch,ch) // push

```

```

5457         ch = '\\\\'
5458     }
5459 }
5460 switch( ch ){
5461     case 'P'-0x40: ch = GO_UP
5462     case 'N'-0x40: ch = GO_DOWN
5463     case 'B'-0x40: ch = GO_LEFT
5464     case 'F'-0x40: ch = GO_RIGHT
5465 }
5466 //fprintf(stderr,"B[%02X]\n",ch);
5467 switch( ch ){
5468     case 0:
5469         continue;
5470
5471     case '\t':
5472         iin.Replace('j');
5473         continue;
5474     case 'X'-0x40:
5475         iin.Replace('j');
5476         continue;
5477
5478     case EV_TIMEOUT:
5479         iin.Redraw();
5480         if iin.pinMode {
5481             fprintf(stderr,"\\J\\r\\n")
5482             iin.inJmode = true
5483         }
5484         continue;
5485     case GO_UP:
5486         if iin.lno == 1 {
5487             continue;
5488         }
5489         cmd.ok := gsh.cmdStringInHistory(iin.lno-1)
5490         if ok {
5491             iin.line = cmd
5492             iin.right = ""
5493             iin.lno = iin.lno - 1
5494         }
5495         iin.Redraw();
5496         continue;
5497     case GO_DOWN:
5498         cmd.ok := gsh.cmdStringInHistory(iin.lno+1)
5499         if ok {
5500             iin.line = cmd
5501             iin.right = ""
5502             iin.lno = iin.lno + 1
5503         }else{
5504             iin.line = ""
5505             iin.right = ""
5506             if iin.lno == iin.lastlno-1 {
5507                 iin.lno = iin.lno + 1
5508             }
5509         }
5510         iin.Redraw();
5511         continue;
5512     case GO_LEFT:
5513         if 0 < len(iin.line) {
5514             xline,tail := delTailChar(iin.line)
5515             iin.line = xline
5516             iin.right = tail + iin.right
5517         }
5518         iin.Redraw();
5519         continue;
5520     case GO_RIGHT:
5521         if( 0 < len(iin.right) && iin.right[0] != 0 ){
5522             xright,head := delHeadChar(iin.right)
5523             iin.right = xright
5524             iin.line += head
5525         }
5526         iin.Redraw();
5527         continue;
5528     case EOF:
5529         goto EXIT;
5530     case 'R'-0x40: // replace
5531         dst := convs(iin.line+iin.right);
5532         iin.line = dst
5533         iin.right = ""
5534         iin.Redraw();
5535         continue;
5536     case 'T'-0x40: // just show the result
5537         readdic();
5538         romkanmode = !romkanmode;
5539         iin.Redraw();
5540         continue;
5541     case 'L'-0x40:
5542         iin.Redraw();
5543         continue;
5544     case 'K'-0x40:
5545         iin.right = "";
5546         iin.Redraw();
5547         continue;
5548     case 'E'-0x40:
5549         iin.line += iin.right
5550         iin.right = "";
5551         iin.Redraw();
5552         continue;
5553     case 'A'-0x40:
5554         iin.right = iin.line + iin.right
5555         iin.line = "";
5556         iin.Redraw();
5557         continue;
5558     case 'U'-0x40:
5559         iin.line = "";
5560         iin.right = "";
5561         iin.clearline();
5562         iin.Redraw();
5563         continue;
5564     case DEL_RIGHT:
5565         if( 0 < len(iin.right) ){
5566             iin.right,_ = delHeadChar(iin.right)
5567             iin.Redraw();
5568         }
5569         continue;
5570     case 0xF: // BS? not DEL
5571         if( 0 < len(iin.line) ){
5572             iin.line,_ = delTailChar(iin.line)
5573             iin.Redraw();
5574         }
5575         /*
5576         else
5577             if( 0 < len(iin.right) ){
5578                 iin.right,_ = delHeadChar(iin.right)
5579                 iin.Redraw();
5580             }

```

```

5581         */
5582         continue;
5583     case 'H'-0x40:
5584         if( 0 < len(iin.line) ){
5585             iin.line,_ = delTailChar(iin.line)
5586             iin.Redraw();
5587         }
5588         continue;
5589     }
5590     if( ch == '\n' || ch == '\r' ){
5591         iin.line += iin.right;
5592         iin.right = "";
5593         iin.Redraw();
5594         fputc(ch,stderr);
5595         AtConsoleLineTop = true
5596         break;
5597     }
5598     if MODE_CapsLock {
5599         if 'a' <= ch && ch <= 'z' {
5600             ch = ch+'A'-'a'
5601         }
5602     }
5603     if MODE_LowerLock {
5604         if 'A' <= ch && ch <= 'z' {
5605             ch = ch+'a'-'A'
5606         }
5607     }
5608     iin.line += string(ch);
5609     iin.Redraw();
5610 }
5611 EXIT:
5612     return iin.line + iin.right;
5613 }
5614
5615 func getline_main(){
5616     line := xgetline(0,"",nil)
5617     fprintf(stderr,"%s\n",line);
5618 /* */
5619     dp = strpbrk(line,"\r\n");
5620     if( dp != NULL ){
5621         *dp = 0;
5622     }
5623
5624     if( 0 ){
5625         fprintf(stderr,"%d\n",int(strlen(line)));
5626     }
5627     if( lseek(3,0,0) == 0 ){
5628         if( romkanmode ){
5629             var buf [8*1024]byte;
5630             convs(line,buf);
5631             strcpy(line,buf);
5632         }
5633         write(3,line,strlen(line));
5634         ftruncate(3,lseek(3,0,SEEK_CUR));
5635         //fprintf(stderr,"outsize=%d\n", (int)lseek(3,0,SEEK_END));
5636         lseek(3,0,SEEK_SET);
5637         close(3);
5638     }else{
5639         fprintf(stderr,"\r\ngetline: ");
5640         trans(line);
5641         //printf("%s\n",line);
5642         printf("\n");
5643     }
5644 */
5645 }
5646 //== end ===== getline
5647
5648 /**
5649 // $USERHOME/.gsh/
5650 //      gsh-rc.txt, or gsh-configure.txt
5651 //      gsh-history.txt
5652 //      gsh-aliases.txt // should be conditional?
5653 //
5654 func (gshCtx *GshContext)gshSetupHomedir()(bool) {
5655     homedir,found := userHomeDir()
5656     if !found {
5657         fmt.Println("--E-- You have no UserHomeDir\n")
5658         return true
5659     }
5660     gshhome := homedir + "/" + GSH_HOME
5661     _,err2 := os.Stat(gshhome)
5662     if err2 != nil {
5663         err3 := os.Mkdir(gshhome,0700)
5664         if err3 != nil {
5665             fmt.Println("--E-- Could not Create $s (%s)\n",
5666                 gshhome,err3)
5667             return true
5668         }
5669         fmt.Println("--I-- Created $s\n",gshhome)
5670     }
5671     gshCtx.GshHomeDir = gshhome
5672     return false
5673 }
5674 func setupGshContext()(GshContext,bool){
5675     gshPA := syscall.ProcAttr {
5676         "", // the starting directory
5677         os.Environ(), // environ[]
5678         []uintptr{os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()},
5679         nil, // OS specific
5680     }
5681     cwd, _ := os.Getwd()
5682     gshCtx := GshContext {
5683         cwd, // Startdir
5684         "", // GetLine
5685         []GChdirHistory { { cwd,time.Now(),0} }, // ChdirHistory
5686         gshPA,
5687         []GCommandHistory{}, //something for invocation?
5688         GCommandHistory{}, // CmdCurrent
5689         false,
5690         []int{},
5691         syscall.Rusage{},
5692         "", // GshHomeDir
5693         Ttid(),
5694         false,
5695         false,
5696         []PluginInfo{},
5697         []string{},
5698         "",
5699         "v",
5700         ValueStack{},
5701         GServer{"","",""}, // LastServer
5702         "", // RSERV
5703         cwd, // RWD
5704         CheckSum{},

```

```

5705     }
5706     err := gshCtx.gshSetupHomedir()
5707     return gshCtx, err
5708 }
5709 func (gsh*GshContext)gshellh(gline string)(bool){
5710     ghist := gsh.CmdCurrent
5711     ghist.WorkDir,_ = os.Getwd()
5712     ghist.WorkDirh = len(gsh.ChdirHistory)-1
5713     //fmt.Printf("--D--ChdirHistory(%d)\n",len(gsh.ChdirHistory))
5714     ghist.StartAt = time.Now()
5715     rusagev1 := Getrusagev()
5716     gsh.CmdCurrent.FoundFile = []string{}
5717     fin := gsh.tgshellh(gline)
5718     rusagev2 := Getrusagev()
5719     ghist.Rusagev = RusageSubv(rusagev2,rusagev1)
5720     ghist.EndAt = time.Now()
5721     ghist.CmdLine = gline
5722     ghist.FoundFile = gsh.CmdCurrent.FoundFile
5723 
5724     /* record it but not show in list by default
5725     if len(gline) == 0 {
5726         continue
5727     }
5728     if gline == "hi" || gline == "history" { // don't record it
5729         continue
5730     }
5731 */
5732     gsh.CommandHistory = append(gsh.CommandHistory, ghist)
5733     return fin
5734 }
5735 // <a name="main">Main loop</a>
5736 func script(gshCtxGiven *GshContext) (_ GshContext) {
5737     gshCtxtBuf,err0 := setupGshContext()
5738     if err0 {
5739         return gshCtxtBuf;
5740     }
5741     gshCtx := &gshCtxtBuf
5742 
5743     //fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
5744     //resmap()
5745 
5746     /*
5747     if false {
5748         gsh_getlinev, with_exgetline :=
5749             which("PATH",[]string{"which","gsh-getline","-s"})
5750         if with_exgetline {
5751             gsh_getlinev[0] = toFullPath(gsh_getlinev[0])
5752             gshCtx.GetLine = toFullPath(gsh_getlinev[0])
5753         }else{
5754             fmt.Println("--W-- No gsh-getline found. Using internal getline.\n");
5755         }
5756     }
5757 */
5758 
5759     ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
5760     gshCtx.CommandHistory = append(gshCtx.CommandHistory,ghist0)
5761 
5762     prevline := ""
5763     skipping := false
5764     for hix := len(gshCtx.CommandHistory); ; {
5765         gline := gshCtx.getline(hix,skipping,prevline)
5766         if skipping {
5767             if strings.Index(gline,"fi") == 0 {
5768                 fmt.Printf("fi\n");
5769                 skipping = false;
5770             }else{
5771                 //fmt.Printf("%s\n",gline);
5772             }
5773             continue
5774         }
5775         if strings.Index(gline,"if") == 0 {
5776             //fmt.Println("--D-- if start: %s\n",gline);
5777             skipping = true;
5778             continue
5779         }
5780         if false {
5781             os.Stdout.Write([]byte("gotline:"))
5782             os.Stdout.Write([]byte(gline))
5783             os.Stdout.Write([]byte("\n"))
5784         }
5785         gline = strsubst(gshCtx,gline,true)
5786         if false {
5787             fmt.Printf("fmt.Printf %v - %v\n",gline)
5788             fmt.Printf("fmt.Printf %s - %s\n",gline)
5789             fmt.Printf("fmt.Printf %x - %s\n",gline)
5790             fmt.Printf("fmt.Printf %U - %s\n",gline)
5791             fmt.Println("Stout.WriteString -")
5792             os.Stdout.Write([]byte(gline))
5793             fmt.Printf("\n")
5794         }
5795     /*
5796     // should be cared in substitution ?
5797     if 0 < len(gline) && gline[0] == '!' {
5798         xline, set, err := searchHistory(gshCtx,gline)
5799         if err {
5800             continue
5801         }
5802         if set {
5803             // set the line in command line editor
5804         }
5805         gline = xline
5806     }
5807     */
5808     fin := gshCtx.gshellh(gline)
5809     if fin {
5810         break;
5811     }
5812     prevline = gline;
5813     hix++;
5814 }
5815 return *gshCtx
5816 }
5817 func main() {
5818     gshCtxtBuf := GshContext{}
5819     gsh := &gshCtxtBuf
5820     argv := os.Args
5821 
5822     if( isin("wss",argv) ){
5823         gj_server(argv[1:]);
5824         return;
5825     }
5826     if( isin("wsc",argv) ){
5827         gj_client(argv[1:]);
5828         return;
5829     }
5830 }

```



```

5953 "CeOBmpqamprbAnjgZ0KamtsCeOBnprpa2prbAnjgaEKa2pga2wJ44GkCmtqa2pqbAnjgaYK"+
5954 "a2tqg2tsCeOBqpgqa2prbAnjgasKa2tr2wJ44GsCmpqg2psCeObRopr2pqg"+
5955 "bnjja4Kamtra2wJ44GvCmpqa2tqbAnjgbIKamtra2wJ44G1CmtsCeOBuApqg2tsCeOBuwpgq"+
5956 "a2tqbAnjgb4Ka2tqg2psCeOBvvpqgbAnjgoAKamtra2psCeOcgQpgq2tqbAnjgbIKCmtqamwJ"+
5957 "44KECmpra2pqbAnjg2Ka2tqg2psCeOCiOpqamtsCeOCipqqa2pqa2wJ44KLcmqg"+
5958 "amwJ44KMCmtqa2psCeCjQpgq2psCeOCjwpramtramsJ44KQCmtqamtarbAnjgpEka2pqaawJ"+
5959 "44KSCmtqa2prbAnjgpHKA2pqa2psCeOdvApra2wJ44KbCmtramprbAnjgpwKa2pramtqbAnj"+
5960 "gIEK";
5961 //</span>
5962
5963 //</span>
5964 /*
5965 <style id="gsh-references-style">
5966 #references details { font-family:Georgia; }
5967 #references a { font-family:Georgia; }
5968 .wrap { white-space: normal; }
5969 </style>
5970 <details id="references"><summary>References</summary><div class="gsh-src">
5971 Web technology
5972 <a href="https://html.spec.whatwg.org">HTML: The Living Standard</a> (September 2020)
5973 <a href="https://html.spec.whatwg.org/dev/">Developer Version</a>
5974
5975 <a href="https://drafts.csswg.org">CSS Working Group Editor Drafts</a> (September 2020)
5976
5977 <a href="https://developer.mozilla.org/ja/docs/Web">MDN web docs</a>
5978 <a href="https://developer.mozilla.org/ja/docs/Web/HTML_Element">HTML</a>
5979 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS">CSS</a> : <span class="wrap">
5980 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Selectors">selectors</a>
5981 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/background-repeat">repeat</a></span>
5982 <a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript">JavaScript</a>
5983 <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP">HTTP</a>
5984 <a href="https://mdn-web-dna.s3-us-west-2.amazonaws.com/MDN-Browser-Compatibility-Report-2020.pdf">MDN Browser Compatibility Report (2020)</a>
5985
5986 Go language (August 2020 / Go 1.15)
5987 <a href="https://golang.org">The Go Programming Language</a>
5988 <a href="https://golang.org/pkg/">Packages</a>
5989 <a href="https://godoc.org/golang.org/x/net/websocket">WebSocket</a>
5990
5991 <a href="https://stackoverflow.com">Stackoverflow</a>
5992 <!--
5993 <iframe src="https://golang.org" width="100%" height="300"></iframe>
5994 -->
5995 </div></details>
5996 */
5997 /*
5998 <details id="html-src" onclick="frame_open();"><summary>Raw Source</summary><div>
5999 <!-- h2>The full of this HTML including the Go code is here.</h2 -->
6000 <details id="gsh-whole-view"><summary>Whole file</summary>
6001 <a name="whole-src-view"></a>
6003 <span id="src-frame"></span><!-- a window to show source code -->
6004 </details>
6005
6006 <details id="gsh-style-frame" onclick="fill_CSSView()"><summary>CSS part</summary>
6007 <a name="style-src-view"></a>
6008 <span id="gsh-style-view"></span>
6009 </details>
6010
6011 <details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
6012 <a name="script-src-view"></a>
6013 <span id="gsh-script-view"></span>
6014 </details>
6015
6016 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Builtin data part</summary>
6017 <a name="gsh-data-frame"></a>
6018 <span id="gsh-data-view"></span>
6019 </details>
6020
6021 </div></details>
6022 */
6023
6024 /*
6025 <div id="GshFooter0"></div>
6026 <!-- 2020-09-17 SatoxITS, visible script { -- >
6027 <details><summary>GJScript</summary>
6028 <style>.gjscript { font-family:Georgia; }</style>
6029 <pre id="gjscript_1" class="gjscript"> function gjtest1(){ alert('Hello GJscript!'); }
6030   gjtest1()
6031 </pre>
6032 <script>
6033   gis = document.getElementById('gjscript_1');
6034   //eval(gis.innerHTML);
6035   //gjs.outerHTML = "";
6036 </script>
6037 </details><!-- ----- END-OF-VISIBLE-PART ----- >
6038 <!--
6039 // 2020-0906 added,
6040 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
6041 https://developer.mozilla.org/en-US/docs/Web/CSS/position
6043 -->
6044 <span id="GshGrid">(^_*)<small>(Hit j k l h)</small></span>
6045
6046 <span id="GStat"><br>
6047 </span>
6048 <span id="GMenu" onclick="GShellMenu(this)"></span>
6049 <span id="GTop"></span>
6050 <div id="GshellPlane" onclick="showGShellPlane();"></div>
6051 <div id="RawTextViewer"></div>
6052 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
6053
6054 <style id="GshStyleDef">
6055 #lineNumbered table,tr,td {
6056   margin:0;
6057   padding:4px;
6058   spacing:0;
6059   border:12px;
6060 }
6061 textarea.LineNumber {
6062   font-size:12px;
6063   font-family:monospace,Courier New;
6064   color:#282;
6065   padding:4px;
6066   text-align:right;
6067 }
6068 textarea.LineNumbered {
6069   font-size:12px;
6070   font-family:monospace,Courier New;
6071   padding:4px;
6072   wrap:off;
6073 }
6074 #RawTextViewer{
6075   z-index:0;
6076   position:fixed; top:0px; left:0px;

```

```

6077     width:100%; xxxheight:50px; xheight:0px;
6078     overflow:auto;
6079     color:#fff; background-color:rgba(128,128,256,0.2);
6080     font-size:12px;
6081     spellcheck:false;
6082   }
6083   #RawTextViewerClose{
6084     z-index:0;
6085     position:fixed; top:-100px; left:-100px;
6086     color:#fff; background-color:rgba(128,128,256,0.2);
6087     font-size:20px; font-family:Georgia;
6088     white-space:pre;
6089   }
6090   #xxxGShellPlane{
6091     z-index:0;
6092     position:fixed; top:0px; left:0px;
6093     width:100%; height:50px;
6094     overflow:auto;
6095     color:#fff; background-color:rgba(128,128,256,0.3);
6096     font-size:12px;
6097   }
6098   #xxxGTop{
6099     z-index:9;
6100     opacity:1.0;
6101     position:fixed; top:0px; left:0px;
6102     width:320px; height:20px;
6103     color:#fff; background-color:rgba(32,32,160,0.15);
6104     color:#fff; font-size:12px;
6105   }
6106   #xxxGPos{
6107     z-index:12;
6108     position:fixed; top:0px; left:0px;
6109     opacity:1.0;
6110     width:640px; height:30px;
6111     color:#fff; background-color:rgba(0,0,0,0.2);
6112     color:#fff; font-size:12px;
6113   }
6114   #GMenu{
6115     z-index:2000;
6116     position:fixed; top:250px; left:0px;
6117     opacity:1.0;
6118     width:100px; height:100px;
6119     color:#fff;
6120     color:#fff; background-color:rgba(0,0,0,0.0);
6121     color:#fff; font-size:16px; font-family:Georgia;
6122     background-repeat:no-repeat;
6123   }
6124   #xxxGStat{
6125     z-index:8;
6126     xopacity:0.0;
6127     position:fixed; top:20px; left:0px;
6128     xwidth:640px;
6129     width:100%; height:90px;
6130     color:#fff; background-color:rgba(0,0,128,0.04);
6131     font-size:20px; font-family:Georgia;
6132   }
6133   #GLog{
6134     z-index:10;
6135     position:fixed; top:50px; left:0px;
6136     opacity:1.0;
6137     width:640px; height:60px;
6138     color:#fff; background-color:rgba(0,0,128,0.10);
6139     font-size:12px;
6140   }
6141   #GshGrid {
6142     z-index:11;
6143     xopacity:0.0;
6144     position:fixed; top:0px; left:0px;
6145     width:320px; height:30px;
6146     color:#9f9; font-size:16px;
6147   }
6148   xbody {display:none;}
6149   .gsh-link{color:green;}
6150   #gsh {border-width:1; margin:0; padding:0;}
6151   #gsh {font-family:monospace,Courier New;color:#ddf;font-size:8px;}
6152   #gsh header{height:100px; background-image:url(GShell-Logo00.png);}
6153   #GshMenu{font-size:14pt;color:#c44;}
6154   .GshMenu{
6155     font-size:14pt;color:#2a2;padding:4px; text-align:right;
6156   }
6157   .GshMenu:hover{
6158     font-size:14pt;color:#fff;font-weight:bold;background-color:#2a2;
6159   }
6160   #GshFooter{height:100px;background-size:80px;background-repeat:no-repeat;}
6161   #gsh note{color:#000;font-size:10pt;}
6162   #gsh h2{color:#24a;font-family:Georgia;font-size:18pt;}
6163   #gsh h3{color:#24a;font-family:Georgia;font-size:16pt;}
6164   #gsh details{color:#888;background-color:#fff;font-family:monospace;}
6165   #gsh summary{font-size:16pt;color:#fff;background-color:#8af;height:30px;}
6166   #gsh summary{font-size:16pt;color:#fff;
6167   xxx-background-color:#8af;
6168   background-color:#6881AD;xxx-PBlue;
6169   height:30px;
6170   }
6171   #gsh pre{font-size:11pt;color:#223;background-color:#faaaffff;}
6172   #gsh a{color:#24a;}
6173   #gsh a[name]{color:#24a;font-size:16pt;}
6174   #gsh .gsh-src{white-space:pre;font-family:monospace,Courier New;font-size:11pt;}
6175   #gsh .gsh-src{background-color:#faaaffff;color:#223;}
6176   #gsh .gsh-src{spellcheck:false}
6177   #SrcTextarea{white-space:pre;font-family:Courier New;font-size:10pt;}
6178   #SrcTextarea{background-color:#faaaffff;color:#223;}
6179   .gsh-code {white-space:pre;font-family:Courier New !important;}
6180   .gsh-code {color:#024;font-size:11pt; background-color:#faaaffff;}
6181   .gsh-golang-data {display:none;}
6182   #gsh-WinId {color:#000;font-size:14pt;}
6183   #gsh-document {font-size:11pt;background-color:#fff;font-family:Georgia;}
6184   .gsh-document {color:#000;background-color:#fff !important;}
6185   .gsh-document > h2{color:#000;background-color:#fff !important;}
6186   .gsh-document details{color:#000;background-color:#fff;font-family:Georgia;}
6187   .gsh-document p{max-width:550pt;color:#000;background-color:#fff;font-family:Georgia;}
6188   .gsh-document address{width:500pt;color:#000;background-color:#fff;font-family:Georgia;}
6189   .gsh-document
6190   @media print {
6191     #gsh pre{font-size:11pt !important;}
6192   }
6193   <!--
6194   // Logo image should be drawn by JavaScript from a meta-font.
6195   // CSS seems not follow line-splittered URL
6196   -->
6197   </style>
6198
6199
6200

```


6449 V1/nmc/cmBw+XnmWlWee82ne1Wo+EkYfUro1IldQp3LawPwRzFp1nHtCoxYq5LcpLQwP
6450 RG4f41bL+uLwZq8zC4FVklLsef1vWn6jg9y+o+*29wB9uZq+jwQWD1l0BwZl9x1v8
6451 KEFL9fdBxp+2Xs6qXkm3dClafTdFadBnUo1uNhnLaFwgB93sevqj1hMULPw/B14a6npG
6452 pksWLw10F6Ymn+3TM16UfbWd3K5Wtsk2zUc0u4js+6Wuy+bt1llPspV1ok327NsJwv
6453 Mg+21W6TWT14T+bxuNDStS719qba20XyD56Dx1uB2RpGvEdrJhD3k13m3zg394VgSp
6454 qzBnk2BwBhBt1eH611/0vpuZgszearL+ntNOCByX90u7q7B6tQ6ubWv/+oy1phw8xh2A4R7
6455 o1Ma0u3q4pYZWHzHwt1a161/0vpuZgszearL+ntNOCByX90u7q7B6tQ6ubWv/+oy1phw8xh2A4R7
6456 9LRATH+Ef5+V0g5flw17STTRMEdA5uQ3t6RyAdm78+/G4B7u1290u9X34An7111S
6457 16Y+xi8sPcmfXrcQyu14KlyH6e9110YHs791/4cxBwngHj2Wbj1LxxJxeQyZuQ0u5Wg5lwdQ
6458 Y6xMfp2XCr6bWtYTp50u72Pu3919rdmN4f5eSbca1ba6stQ07+be1ubSppBmrhC27B1
6459 0YQhs10vJcLk1Y4fxKo2+kzWp+oja1DvsEvg9Er9Phe+b+XwNxMkM6lVpQnUiXk1zxn+OpveQq
6460 h4F8J1J9wOrt=645t+50002zdG5+CdTsP3/VXH1nqagQ1+AB2j8s8ss/Fm19D3cmjwbo
6461 kRn3kxzquzp2sKzUk1bc0rnPh1NaB1Ms135d1J1r9y0wv9Np1k9lq19C93Idm7v2
6462 8/kpgnxKvc1bdle1DD0tUos+rxmsDipnxm7w81YMeH2GzD2tfgJn+>xzusBpV04zrhBw9
6463 H926+8JSVHoszRdi171AAp0WQz17LspLeplBQ9oyXbyb8/8dm2+//11/qnago2awPc38/3WEl
6464 I4aa5f05EMXARkao12CCp2xJNv+1M72Lkh3v27lwv7n9w4/+6jgkxJLplq7b1tADKwlp/
6465 n1hs+QSi+HiEw5PruVeng20d6nf7Kto1f/d1kuasCeeB1AEB1AEB1AEB1AEB1AEB1AEB1A
6466 E1EB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1A
6467 E1EB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1A
6468 E1EB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1A
6469 E1EB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1A
6470 E1EB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1AEB1A
6471 ITSMore="data:image/png;base64,"
6472 iVBORw0KGgoAAAANSUhEUgAAEAAAABACAYAAQcGxAhEAAAABGdTUEAAALGPC/xhBQAACBj"
6473 Hk1LEQ9v4jdxTsA2EWyGCMYX7k1CkVgjxvaBe7CrAxs1d1Aw4SgW5hM5zEVs+SgS+ZBQ
6474 8gbcdBfHyzw8sMszaUBHn=MkAd4C8LpD8npg47UpCgkI2j18IFx3EwPlWhNkVhyWecev
6475 UebDxbAx02XaujheYDQnK1QasspCjkj4nW31fswgY6jKu6/PvAhPh0alSfVhe70kdwMrw
6476 yMGSSuPyWwah19ktKv2s3d2nRu2CpW8gg4Rpa1L9sJp9vCp1nR4DxPKin8xaQciT6Lzq
6477 Z0Rdhw/4+u2Gzq1S8gbqbgvMkrfr1n6XK80ql0d00m1GtMv2PERA8L9vbfobPscL33sVtyr
6478 S9wiQdzhuu13v8n783/gBuUs2elg1c8gAAABJRU5ErkJggg=";

6479

6480 ConfigIcon_Data="data:image/png;base64,"
6481 "1VBORw0KGgoAAAANSUhEugAAAAAAAACBACAYAAQcGxAhEAAAABGdTUEAAALGPC/xhBQAACBj"
6482 "SFJNAb67gAgQ1AOpaAACAGAA6dTAao9g+A6AmfaAMC3U1c188AahayGVSYWNTQAgAAAa
6483 "CAAPFARIAAAAQAAQARoAQAoBAAAEEAAEAKArBQAABAAAEEAAEAAASASAgAAWAAAEEAAhAgA2kA
6484 "BAAAEEAAAABAAAQAAQAAEAAAQAAEAAAQAAEAAAQAAEAAAQAAEAAAQAAEAAAQAAEAAAQAAEAAA
6485 "QKADAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAAQAAA
6486 "Q29+L9Lkb2J1LnhhtcAAAANAAAAPHg6e1cbwvYOBs4Bwxcuzp58PsJh2911TzPucp2zXHrLy1g"
6487 "edb+4pxB0a0z1oWE10InemvLghN4S0Ljai1pogCIAcRm0l1ER18B4wXuwpzcyG9VhImhd0A
6488 "Ly93d3cudzub3JnL250TkvMd1JmtchmLn5hBnRheC1m9p1kqigCAGiCAGiCAGiCAGiCAGiCAG
6489 "bn'MuYrWvYmuYmU29tL3RpZm1vNs4wLh1YIKC1AgCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAG
6490 "L25zLkfB2J1LnnLvb9s1Gz1Lmz1Gw10f0gGicAg1CAGiCAGiCAGiCAGiCAGiCAGiCAGiCAG
6491 "PC90a0wMk0mkyawGdf0wauP9ogicAg1CAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAG
6492 "aWzMo1L29x2dRzbv5Bm10f0g9icAg1CAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAG
6493 "Q29+5b3jTcfGp1T42ICAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAG
6494 "aXh1bFeW11bnBp2+ciAg1CAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAG
6495 "aWY6U142ZwXzRgl2TzWz2a9WuP9ogicAg1CAGiCAGiCAGiCAGiCAGiCAGiCAGiCAGiCAG
6496 "UkRgRp0L83g6e1cbwvYb0T4KsN1QaaLup1JREFUewHf314VFWV/1L313eyyGvCsdGardE
6497 "lcw1Lx2Hu51x90NzG31s5oPe/2x3a2v82pb+6dHe+2wzNtoPxatPjWVEERQwsBdWl01Rd
6498 "lwhSS3v1+9pJfJKC0mte13xV3yK999z3zNc2p5b3h0lry1pHt0zNp0QdWvB
6499 "CH+Ce/Ne124Hx4VY/vfxRKh31pwoCoc/fcd4p0y7u0J7oWtJ+nWmSH3RiXpQyJzA9jS9hF
6500 "23uH/plus1/A11b1KPxSees86mrt7UdfTrRwf1zLkW8c1yXgMcawSz521uCgkL4ySS1Hq+L
6501 "wurePhnM0n/qdkHf0NbmXkVt6y15QShy+H01S/C1mfPxT3YD94k9Eka1uLh1nB1z82vUATxZ
6502 "MD4UhRkRyEj6QJhp1OJiQhEv4aQmduo6WkR4N6Vw+qGrpQdJe5sZubKvvgY0c14wLQtzr
6503 "h1t3BSGNgbr1l85Rg7z2h3tbsjcsf0uFksXnBbVgXy1e06Fy1drbkhE85j1n2k8rg+io
6504 "U1RJLJUyAaUPsBxKmoRng1RtWxGKxj0e8pJ7d7zr7Hmz8fHdEbnLw4Ew14HxZy7d9y6
6505 "pgx4P2x4Bxan1u/x00Q91Ax1y28BwF62ziy1PbHr1z1yE1F7L25JLwPf2dWzvB
6506 "YmpCkxBQhkb2gvAn24P9y12z8rCwAOw6gywv1HvriuHcOKxHedSOBOhWrrBy/AGTEwhz
6507 "orkXhRgj75emad4cqMsFy1GmMgnB9rxFyp02fJUpBdpj0m6rbCgj707uY1mkQ1kEab
6508 "p"UetXckwpk4s1eYt2E10x5mFXYt0zko1jY7v5sSg5045qH+fKuQhJp0aa7p4oav1Cw
6509 "isD7de57Gz4NFRHk1b5QShy+H01S/C1mfPxT3YD94k9Eka1uLh1nB1z82vUATxZ
6510 "SkdeA1PhGm1TsEtdTMOQkowTu24z7Ag1J4L21TupUqhnqia1Cq1e70p2zDq1LjvCaje1Y
6511 "muorKhwdc6G6H6G6F6G40352R+H+lmxDwSx5tB2p5h0lry1pHt0zNp0QdWvB
6512 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6513 "aBkMm4Fd2p1dnL111o7q2ScDn111q3zQJ2u01bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6514 "tGhY1fJg13nRq3pW60v15RMQXTEStf1z2TgxQd0nT1XuVeao8zC15T16f1R11B2+Zw
6515 "Pl2y3dsSkwN11iF0ctqjnyH04Nk9Ms1dVb9s1m2nBzHdVvYxvhmB1z82vUATxZ
6516 "3KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6517 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6518 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6519 "5Vmhn1YiQD9M97u1sJe6A1GcLgkbed/GtCsHx66d4iOrJFJxgyFmaoYHwYgkHx0g+cqy/c
6520 "abKmM4Fd2p1dnL111o7q2ScDn111q3zQJ2u01bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6521 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6522 "+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6523 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6524 "nrmJwL10i2z0B3k51gk1t7y19+K0MgB6vKt7RahBgfmlw185H0GciWxD5
6525 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6526 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6527 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6528 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6529 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6530 "zcs5Vx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6531 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6532 "oy4Y4rJyXwLkFtrk2Bd5x2zS6b1y74u5q2mDev/yx10000000000000000000000000000000
6533 "6g751/SMA3nisCpkSa2u0f5z3n9t3u1yf15v0y5HkmoG3nOC30nAl33aLm3n1aM3i2mz7K
6534 "4z2YoVQFW72s3b2z0B2z0xR1p2Y4R03L0H4d7uCa70vC0u1f0p0d1p61g
6535 "lNywg6b062234T0Fpxvdi1uW1xUibxHes7G5M2H7Df0H7y1l1t7891z2w19tbrNnsLwju5
6536 "557KmXarX+yH06y6wNrnL1UcM0Lkroj1C2a1stvM7p1Y6rB4Dw1tRr0d0y20zNcny1
6537 "v+3D04mxKmsVx0bHg2z2nRgeyW2uNN47H88BgePz0tQaqEdGOD4j13rsJLHtDj2d4g
6538 "Kfc+x10B91m5XzNq2Bp8WnBp2CttxhdwLz83l1by6dufx8FhxusC15d2Q4jx/Aj09b2x+gH
6539 "i3Py/01Huclw1snJy1h9n0+3L/gm1fkWg8/s1Qy10n4p9vPpE0/objy1CayVaFqTz2T
6540 "e20pmzE11TGSRLxer0e0utpjS1WtCkD0s140l2yMsdo2E5p0d5TfPw12l0Tpkby1s0d1ls
6541 "zcs30216p16t2p1QrJn06L62g1Bz1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6542 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6543 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6544 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6545 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6546 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6547 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6548 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6549 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6550 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6551 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6552 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6553 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6554 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6555 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6556 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6557 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6558 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6559 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6560 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6561 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6562 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6563 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6564 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6565 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6566 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6567 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6568 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6569 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6570 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6571 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6572 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6573 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6574 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6575 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6576 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6577 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6578 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6579 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6580 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6581 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6582 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6583 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6584 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6585 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6586 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6587 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6588 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6589 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6590 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6591 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6592 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6593 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6594 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6595 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6596 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6597 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6598 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6599 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6600 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6601 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6602 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6603 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6604 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6605 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6606 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6607 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6608 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6609 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6610 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6611 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6612 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6613 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6614 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6615 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6616 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6617 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6618 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6619 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6620 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6621 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6622 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6623 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6624 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6625 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6626 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6627 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6628 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6629 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6630 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6631 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6632 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6633 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6634 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6635 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g9y+L
6636 "gx2wKkGuX4vh1g6y17L2d3g3v9UyWzQAcgtMUR3xRv2D9W/+F0vRQmguJ7Cg20YKmW
6637 "33KxXyDcorQcN1lP0lRadq1YegTpyR6U3pWbBDSs+vBb1gVxTCzOkyCzK9n3x9
6638 "P1QjbA/BVR95p6hVcHuAeAcDh2zA2k7D1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6639 "B1PgxGX1KgcFu01oYjazw7unpxQm9x1e7MwL2o1bz1zue1lTmR0c4d7d4+/fxf/dxAmxy
6640 "8ex1uNqAtyrQdPfIker7Aq1bQ7kD1zJ1zKj2u1yJevCt2TS2y5W1zQsfCgdbWkUbt
6641 "52+ta2zb+j3nH2puy1jhs7R2x8BnHt1720xV4JwHd56f1zj0k1+Qz1oKv+C+T2z+CuHg
6642 "ptNo4wvpxBzAu17F4B1Ysgf148VbGv+R3dC0eSg1515Hs15Shx1Su61g6eDooQ9nK9xS
6643 "30MKQCoVpe/hJamaCwubaoX-AueyV1lk80h2CyzKp1Jy7v140LbG+6d+LwzNtoPxatPjW
6644 "d3PfHcs0KhdLx1zR2Cn17XzJ0dx2603q39w2Am2dole2Xqno79vCf7MpdC1d19y0d74p
6645 "wuzC2zSpkn4yzD9D1KAvaP9enj/RidAn351o5d4eP0B9h8nCkxj1Jbhsa0jhwd3f1r1bhc
6646 "e9kX7b/9a193g6m6qNrcNE9GtMfTpvsre1pVmeH72Xg6gUTs+TzE6C1b1p0f0d1p61g
6647 "yKU1RaH91y9jowPfLcgk1wass550/PfB4x10K3x/Dpm9p19f1c2f41z7KSE15Pvtfdxgys
6648 "53zSVx+8URVzq597u7BomtYuumXh1Cidn1a1Opf9j1R1m7g

```

6573 "SoNT3eUZYdyhMsDp27xY9YRHEIirH+dl6Q18XPag/tWpBubplav83zAqQ4dPOH2bfXfzp+"
6574 "WjYhi01/LacjBEzNcid/rJ61vzCnhey821mkLGENIVYqs+rohRaM7Knu7DumGsyAyLNARlmg"+
6575 "qzZqOxjPUULjy8SSGbGanC23GmeAvZNmx18cacAGngcUlpsVBBehXwAsBbJcuuoP8Hj5g09P+"
6576 "mrzrXXpqDF6C7qeOF7bz6m9pdBP8Xlxw7/7AW9U0IT1cVptt0L3K9E51VFdtxgu890GDgx64+"
6577 "J0+2qdFmo/err5cxjYEff7zqf4AAVCahKEYlf*Llk6jhYYPsOH74vUKEhfF0mBeZoK/jAq19+"
6578 "KsPT/2nuUhN0sTw5yljpxoElvVMdwvEaByJLJoG7vua3650guzx8w7d/YW16nv2xB0vMW+"
6579 "A4/3B7FD/gruPrltCOF2V42dc1jGPkXdxbhYKtyexuZGBNdprLmpla575yeucc18H1KX+"+
6580 "TnEqey7qjxUQkvbfu0H6Lxlvu2afNPZ4cbc085bhqmMpDkkZocGKAh61/40rw8raTpq5+"
6581 "MIKx+aai7gxQxyemAk51s6yup3j/eUohJFS6qvXwypHuV6d14VN+f+e/PScAAa6chjkB+"+
6582 "1XXteIJhZX1Rwd5hsAbkzcZbp5//SLZx7jxX0XD13HW/FXYRDd0mV4z9WmyBCHRSCLPFz+"+
6583 "r50w670R/cg97u8T8tEq/tvn2GSOJWR/vb2h1D05GyTxj9Mx0MKwAw5fxRE+"+
6584 "uColPLe8cBzbfekeymizivttvwE13M85nhJyfTlkh3jy8N04x4Or9HGkntVd8WQNAp0Mr+1BdT+"+
6585 "TAH86UbD66Ygk9N8uk2yL7NHD21XD0yyGDUEkMh16pcCxxEs5d4mB37/d/km0bEKQ3+"+
6586 "6zeOYcoOBjPNBInUew8g61mlNL/152pjpTUTRJ+rpa47dxB9fxMnxLMSx0TIZbjG/Wgrpn1a+"+
6587 "25ktAwSgVcf3vUDQVFlpjrlXcn69Ajx0fxD6bnJyFBynL+GHebeBO/JmpacrncDjkTKxz+"+
6588 "fivcALz+D1Jdr/sfl+5HA42DrmmP0JhlxF4e/m14lJmPdaAx03PtMCQ5UTCOPTp3qd6el+"+
6589 "6gZVAAjyaJViCr/s78jnjo02jirqdBj9SvsmJ2EUHc3xj7a2nx/HawqJifhjOr13AhihEpi+"+
6590 "WiIseaFKNw+n0dzmK3YU0u1j0NL5pnfaJlh3y8N04x4Or9HGkntVd8WQNAp0Mr+1BdT+"+
6591 "4eg8ppjcx1SxgXmzzLio/zouNuLNpVXmewFtry/gkpzCcrHxHazlJpdV/+6rq5BvmPrnhMx+"+
6592 "01kr5xjyXc3T660Mu942Jdp8L7B0xrrRQHVmcGd37915vdur80g7h0UM21V4+tSVjaD3xQY+"+
6593 "sw#B0uOafcoZ2MP184YXJuXupplai2w6JdrPhrdTEp47qd+5nx1cf9QbwNps7ooteley6G+"+
6594 "V6A+5au3KXB1fhbQU84j425K464aJkLKSX8YKThr7hobswILYxiK3X3g+0V2aZzY8a5+mLE+"+
6595 "YoePlJcVPskdgi912m+macdkGGgwBBDh/KfN8M1Z2kOMOJ1vhggdfd8MsV4B6yfQTvVO+"+
6596 "v/3NDR+MzSV3wz1pZGxsasGL3+iLLAL+6owv8da8M5KgMgxNsBrjfC152A2x7H8JQWb4+"+
6597 "FffdrRTE33AtvnRuMxMzLy8tOWQkupjByVhdibl+L4as06whnhlegq3dOnPxufystZbqu+"+
6598 "598/f2FyH1xj0QWmvtbdK3+dVYH3W+GSN8yLlvyaXpi06vyde7Zfo85tnfdWBqlg8Ewk4t10/zA+"+
6599 "LsyzfIEeBmdrGzvNVor7bpAahUmrikykdY+mq655Ls8Lvg6NiAYQs1tTnGzBNXoeCkCPYQg+"+
6600 "Xcb0tRx+TcNQIteVjilaCY/HOWWVxshEV7n4maojtOpqpfjjHoYx2Y89zgSewFH044r+"+
6601 "i74dgwvNWXYSMR6icArOojsMSPa0P99F5UiT3R6ned9DAZBWTCThyt0Tn6MxkuIkP+tuImzxxq+"+
6602 "tCwrgjEjxGKwXBrJRI6uvMtYgsu03aijQtGx328K/CTC8mo689CT5Pbtl+JLZayikk+g9B+"+
6603 "f+2e7tu4L/VV1En8ShhhNF4R/BKiajfDybIVVT6x6Dhs0yKbXWJV651FjdY+rzoQ/USKAY+"+
6604 "Xwcd/s3LIx6sPrtCFIV3UH1Crtyps/n5/BCUf1Hb5/P/D94D1z5tiuE3AAAEE1FTkSu+"+
6605 "Qmc";
6606 "</script>
6607
6608 <div id="GJFactory_1" class="xxxGJFactory" style=""></div>
6609 <!--
6610 https://developer.mozilla.org/en-US/docs/Web/CSS/line-height
6611 -->
6612 <style>
6613 .GJFactory{
6614   resize:both; overflow:scroll;
6615   position:static;
6616   border:1.2px dashed #282; border-radius:2px;
6617   margin:0px; padding:10px !important;
6618   width:340px; height:340px;
6619   flex-wrap: wrap;
6620   color:#fff; background-color:rgba(0,0,0,0);
6621   line-height:0.0;
6622   xxxcolor:#22a !important;
6623   text-shadow:2px 2px #ddf;
6624 }
6625 .GJFactory h1,h2,h3,h4 {
6626   xxxcolor:#22a !important;
6627 }
6628 .xxxinput {
6629   border:1px dashed #0f0; border-radius:0px;
6630 }
6631 .GJWin:hover{
6632   color:#df8 !important;
6633   background-color:rgba(32,32,160,0.8) !important;
6634   line-height:0.0;
6635 }
6636 .GJWin:active{
6637   color:#df8 !important;
6638   background-color:rgba(224,32,32,0.8) !important;
6639   line-height:0.0;
6640 }
6641 .GJWin:focus{
6642   color:#df8 !important;
6643   background-color:rgba(32,32,32,1.0) !important;
6644   line-height:0.0;
6645 }
6646 .GJWin{
6647   z-index:10000;
6648   display:inline;
6649   position:relative;
6650   flex-wrap: wrap;
6651   top:0; left:0px;
6652   width:285px !important; height:205px !important;
6653   border:1px solid #eaa; border-radius:2px;
6654   margin:0px; padding:0px;
6655   font-size:8pt;
6656   line-height:0.0;
6657   color:#fff; background-color:rgba(0,0,64,0.1) !important;
6658 }
6659 .GJTab{
6660   display:inline;
6661   position:relative;
6662   top:0px; left:0px;
6663   margin:0px; padding:2px;
6664   border:0px solid #000; border-radius:2px;
6665   width:90px; height:20px;
6666   font-family:Georgia;
6667   font-size:9pt;
6668   line-height:1.0;
6669   white-space:nowrap;
6670   color:#fff; background-color:rgba(0,0,64,0.7);
6671   text-align:center;
6672   vertical-align:middle;
6673 }
6674 .GJStat:focus{
6675   color:#df8 !important;
6676   background-color:rgba(32,32,32,1.0) !important;
6677   line-height:1.0;
6678 }
6679 .GJStat{
6680   display:inline;
6681   position:relative;
6682   top:0px; left:0px;
6683   margin:0px; padding:2px;
6684   border:0px solid #0f0; border-radius:2px;
6685   width:160px; height:20px;
6686   font-family:monospace;
6687   font-size:9pt;
6688   line-height:1.0;
6689   color:#fff; background-color:rgba(0,0,64,0.2);
6690   text-align:center;
6691   vertical-align:middle;
6692 }
6693 .GJIcon{
6694   display:inline;
6695   position:relative;
6696   top:0px; left:1px;

```

```
6697     border:2px solid #44a;
6698     margin:0px; padding:1px;
6699     width:13.2; height:13.2px;
6700     border-radius:2px;
6701     font-family:Georgia;
6702     font-size:13.2px;
6703     line-height:1.0;
6704     white-space:nowrap;
6705     color:#fff; background-color:rgba(32,32,160,0.8);
6706     text-align:center;
6707     vertical-align:middle;
6708     text-shadow:0px 0px;
6709   }
6710   .GJText:focus{
6711     color:#fff !important;
6712     background-color:rgba(32,32,160,0.8) !important;
6713     line-height:1.0;
6714   }
6715   .GJText{
6716     display:inline;
6717     position:relative;
6718     top:0px; left:0px;
6719     border:0px solid #000; margin:0px; padding:0px;
6720     width:280px; height:160px;
6721     border:0px;
6722     font-family:Courier New,monospace !important;
6723     font-size:8pt;
6724     line-height:1.0;
6725     white-space:pre;
6726     color:#fff; background-color:rgba(0,0,64,0.5);
6727     background-color:rgba(32,32,128,0.8) !important;
6728   }
6729   .GJMode{
6730     display:inline;
6731     position:relative;
6732     top:0px; left:0px;
6733     border:0px solid #000; border-radius:0px;
6734     margin:0px; padding:0px;
6735     width:280px; height:20px;
6736     font-size:9pt;
6737     line-height:1.0;
6738     white-space:nowrap;
6739     color:#fff; background-color:rgba(0,0,64,0.7);
6740     text-align:left;
6741     vertical-align:middle;
6742   }
6743 </style>
6744
6745 <script id="gsh-script">
6746 // 2020-0909 added, permanent local storage
6747 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
6748 var MyHistory = ""
6749 Permanent = localStorage;
6750 MyHistory = Permanent.getItem('MyHistory')
6751 if( MyHistory == null ){ MyHistory = "" }
6752 d = new Date()
6753 MyHistory = d.getTime()/1000+" "+document.URL+"\n" + MyHistory
6754 Permanent.setItem('MyHistory',MyHistory)
6755 //Permanent.setItem('MyWindow',window)
6756
6757 var GJLog_Win = null
6758 var GJLog_Tab = null
6759 var GJLog_Stat = null
6760 var GJLog_Text = null
6761 var GJWin_Mode = null
6762 var FProductInterval = 0
6763
6764 var GJ_FactoryID = -1
6765 var GJFactory = null
6766 if( e = document.getElementById('GJFactory_0') ){
6767   GJFactory_1.height = 0
6768   GJFactory = e
6769   e.setAttribute('class','GJFactory')
6770   var GJ_FactoryID = 0
6771 }else{
6772   GJFactory = GJFactory_1
6773   var GJ_FactoryID = 1
6774 }
6775
6776 function GJFactory_Destroy(){
6777   gif = GJFactory
6778   //gif = document.getElementById('GJFactory')
6779   //alert('gif='+gif)
6780   if( gif != null ){
6781     if( gif.childNodes != null ){
6782       for( i = 0; i < gif.childNodes.length; i++ ){
6783         gif.removeChild(gif.childNodes[i])
6784       }
6785     }
6786     gif.innerHTML = ''
6787     gif.style.width = 0
6788     gif.style.height = 0
6789     gif.removeAttribute('style')
6790   GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
6791   window.clearInterval(FProductInterval)
6792   return '-- Destroy: work product destroyed'
6793 }else{
6794   return '-- Destroy: work product not exist'
6795 }
6796 }
6797
6798 var TransMode = false
6799 var OnKeyControl = false
6800 var OnKeyShift = false
6801 var OnKeyAlt = false
6802 var OnKeyJ = false
6803 var OnKeyK = false
6804 var OnKeyL = false
6805
6806 function GJWin_OnKeyUp(ev){
6807   keycode = ev.code;
6808   if( keycode == 'ShiftLeft' ){
6809     OnKeyShift = false
6810   }else
6811   if( keycode == 'ControlLeft' ){
6812     OnKeyControl = false
6813   }else
6814   if( keycode == 'AltLeft' ){
6815     OnKeyAlt = false
6816   }else
6817   if( keycode == 'KeyJ' ){ OnKeyJ = false }else
6818   if( keycode == 'KeyK' ){ OnKeyK = false }else
6819   if( keycode == 'KeyL' ){ OnKeyL = false }else
6820 }
```

```

6821     }
6822     ev.preventDefault()
6823   }
6824   function and(a,b){ if(a){ if(b){ return true; } return false; } }
6825   function GJWin_OnKeyDown(ev){
6826     keycode = ev.code;
6827     mode = '';
6828     key = '';
6829     if( keycode == 'ControlLeft' ){
6830       OnKeyControl = true
6831       ev.preventDefault()
6832       return;
6833     }else
6834     if( keycode == 'ShiftLeft' ){
6835       OnKeyShift = true
6836       ev.preventDefault()
6837       return;
6838     }else
6839     if( keycode == 'AltLeft' ){
6840       ev.preventDefault()
6841       OnKeyAlt = true
6842       return;
6843     }else
6844     if( keycode == 'Backquote' ){
6845       TransMode = !TransMode
6846       ev.preventDefault()
6847     }else
6848     if( and(keycode == 'Space', OnKeyShift) ){
6849       TransMode = !TransMode
6850       ev.preventDefault()
6851     }else
6852     if( keycode == 'ShiftRight' ){
6853       TransMode = !TransMode
6854     }else
6855     if( keycode == 'Escape' ){
6856       TransMode = true
6857       ev.preventDefault()
6858     }else
6859     if( keycode == 'Enter' ){
6860       TransMode = false
6861       //ev.preventDefault()
6862     }
6863     if( keycode == 'KeyJ' ){ OnKeyJ = true }else
6864     if( keycode == 'KeyK' ){ OnKeyK = true }else
6865     if( keycode == 'KeyL' ){ OnKeyL = true }else
6866     {
6867     }
6868     if( ev.altKey ){ key += 'Alt+' }
6869     if( onKeyControl ){ key += 'Ctrl+' }
6870     if( OnKeyShift ){ key += 'Shift+' }
6871     if( and(keycode != 'KeyJ', OnKeyJ) ){ key += 'J+' }
6872     if( and(keycode != 'KeyK', OnKeyK) ){ key += 'K+' }
6873     if( and(keycode != 'KeyL', OnKeyL) ){ key += 'L+' }
6874     key += keycode
6875
6876     if( TransMode ){
6877       //mode = "[\u343\201\202r]"
6878       mode = "[\u{343}\u{201}\u{202r}]"
6879     }else{
6880       mode = '[---]'
6881     }
6882   }
6883   //// /gjmode.innerHTML = "[---]"
6884   GJWin_Mode.innerHTML = mode + ' ' + key
6885   //alert('Key:' +keycode)
6886   ev.stopPropagation()
6887   //ev.preventDefault()
6888 }
6889 function GJWin_OnScroll(ev){
6890   x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
6891   y = DragStartY = gsh.getBoundingClientRect().top.toFixed(0)
6892   GJLog_append('OnScroll: x=' +x+ ',y=' +y)
6893 }
6894 document.addEventListener('scroll',GJWin_OnScroll)
6895 function GJWin_OnResize(ev){
6896   w = window.innerWidth
6897   h = window.innerHeight
6898   GJLog_append('OnResize: w=' +w+ ',h=' +h)
6899 }
6900 window.addEventListener('resize',GJWin_OnResize)
6901
6902 var DragStartX = 0
6903 var DragStartY = 0
6904 function GJWin_DragStart(ev){
6905   // maybe this is the grabbing position
6906   this.style.position = 'fixed'
6907   x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
6908   y = DragStartY = this.getBoundingClientRect().top.toFixed(0)
6909   GJLog_Stat.value = 'DragStart: x=' +x+',y=' +y
6910 }
6911 function GJWin_Drag(ev){
6912   x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
6913   this.style.left = x - DragStartX
6914   this.style.top = y - DragStartY
6915   this.style.zIndex = '30000'
6916   this.style.position = 'fixed'
6917   x = this.getBoundingClientRect().left.toFixed(0)
6918   y = this.getBoundingClientRect().top.toFixed(0)
6919   GJLog_Stat.value = 'x=' +x+',y=' +y
6920   ev.preventDefault()
6921   ev.stopPropagation()
6922 }
6923 function GJWin_DragEnd(ev){
6924   x = ev.clientX; y = ev.clientY
6925   //x = ev.pageX; y = ev.pageY
6926   this.style.left = x - DragStartX
6927   this.style.top = y - DragStartY
6928   this.style.zIndex = '30000'
6929   this.style.position = 'fixed'
6930   if( true ){
6931     console.log("Dropped: "+this.nodeName+'#'+this.id+' x=' +x+ ' y=' +y
6932     +' parent=' +this.parentNode.id)
6933   }
6934   x = this.getBoundingClientRect().left.toFixed(0)
6935   y = this.getBoundingClientRect().top.toFixed(0)
6936   GJLog_Stat.value = 'x=' +x+',y=' +y
6937   ev.preventDefault()
6938   ev.stopPropagation()
6939 }
6940 function GJWin_DragIgnore(ev){
6941   ev.preventDefault()
6942   ev.stopPropagation()
6943 }
// 2020-09-15 let every object have console view!

```

```

6945 var GJ_ConsoleID = 0
6946 var PrevReport = new Date()
6947 function GJLog_StatUpdate(){
6948   txa = GJLog_Stat;
6949   if( txa == null ){
6950     return;
6951   }
6952   tmLap0 = new Date();
6953   p = txa.parentNode;
6954   pw = txa.getBoundingClientRect().width;
6955   ph = txa.getBoundingClientRect().height;
6956   //txa.value += '#'+p.id+' pw=' +pw+', ph=' +ph+'\n';
6957   tx1 = '#'+p.id+' pw=' +pw+', ph=' +ph+'\n';
6958
6959   w = txa.getBoundingClientRect().width;
6960   h = txa.getBoundingClientRect().height;
6961   //txa.value += 'w=' +w+', h=' +h+'\n';
6962   tx1 += 'w=' +w+', h=' +h+'\n';
6963
6964   //txa.value += '\n';
6965   //txa.value += DateShort() + '\n';
6966   tx1 += '\n';
6967   tx1 += DateShort() + '\n';
6968   tmLap1 = new Date();
6969
6970   txa.value += tx1;
6971   tmLap2 = new Date();
6972
6973   // vertical centering of the last line
6974   sHeight = txa.scrollHeight - 30; // depends on the font-size
6975   tmLap3 = new Date();
6976
6977   txa.scrollTop = sHeight; // depends on the font-size
6978   tmLap4 = new Date();
6979
6980   now = tmLap0.getTime();
6981   if( PrevReport == 0 || 10000 <= now-PrevReport ){
6982     PrevReport = now;
6983     console.log('StatusBarUpdate:',
6984       + 'leng=' + txa.value.length + ' byte,' +
6985       + 'time=' + (tmLap4 - tmLap0) + 'ms {' +
6986       + 'tadd=' + (tmLap2 - tmLap1) + ', ' +
6987       + 'hcal=' + (tmLap3 - tmLap2) + ', ' +
6988       + 'scrl=' + (tmLap4 - tmLap3) + '}';
6989   );
6990 }
6991
6992 GJWin_StatUpdate = GJLog_StatUpdate;
6993 function GJ_showTimel(wid){
6994   //e = document.getElementById(wid);
6995   //console.log(wid.id+'.value.length=' +wid.value.length)
6996   if( e != null ){
6997     //e.value = DateShort();
6998   }else{
6999     // should remove the Listener
7000   }
7001 }
7002 function GJWin_OnResizeTextarea(ev){
7003   this.value += 'resized:' + '\n';
7004 }
7005 function GJ_NewConsole(wname){
7006   wid = wname + '_' + GJ_ConsoleID
7007   GJ_ConsoleID += 1
7008
7009   GJFactory.style.setProperty('width',360+'px'); //GJFsize
7010   GJFactory.style.setProperty('height',320+'px')
7011   e = GJFactory;
7012   console.log('GJFa #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
7013
7014   if( GJFactory.innerHTML == "" ){
7015     GJFactory.innerHTML = '<' + 'H3>GJ Factory_' + GJ_FactoryID + '<' + '/' + 'H3><' + 'hr>\n';
7016   }else{
7017     GJFactory.innerHTML += '<' + 'hr>\n';
7018   }
7019
7020   gjwin = GJLog_Win = document.createElement('span')
7021   gjwin.id = wid
7022   gjwin.setAttribute('class','GJWin')
7023   gjwin.setAttribute('draggable','true')
7024   gjwin.addEventListener('dragstart',GJWin_DragStart)
7025   gjwin.addEventListener('drag',GJWin_Drag)
7026   gjwin.addEventListener('dragend',GJWin_Drag)
7027   gjwin.addEventListener('dragover',GJWin_DragIgnore)
7028   gjwin.addEventListener('dragenter',GJWin_DragIgnore)
7029   gjwin.addEventListener('dragleave',GJWin_DragIgnore)
7030   gjwin.addEventListener('dragexit',GJWin_DragIgnore)
7031   gjwin.addEventListener('drop',GJWin_DragIgnore)
7032   gjwin.addEventListener('keydown',GJWin_OnKeyDown)
7033
7034   gjtab = GJLog_Tab = document.createElement('textarea')
7035   gjtab.addEventListener('keydown',GJWin_OnKeyDown)
7036   gjtab.style.readonly = true
7037   gjtab.contenteditable = false
7038   gjtab.value = wid
7039   gjtab.id = wid + '_Tab'
7040   gjtab.setAttribute('class','GJTab')
7041   gjtab.setAttribute('spellcheck','false')
7042   gjwin.appendChild(gjtab)
7043
7044   gjstat = GJLog_Stat = document.createElement('textarea')
7045   gjstat.addEventListener('keydown',GJWin_OnKeyDown)
7046   gjstat.id = wid + '_Stat'
7047   gjstat.value = DateShort()
7048   gjstat.setAttribute('class','GJStat')
7049   gjstat.setAttribute('spellcheck','false')
7050   gjwin.appendChild(gjstat)
7051
7052   gjicon = document.createElement('span')
7053   gjicon.addEventListener('keydown',GJWin_OnKeyDown)
7054   gjicon.id = wid + '_Icon'
7055   gjicon.innerHTML = '<font color="#f44">J</font>'
7056   gjicon.setAttribute('class','GJIcon')
7057   gjicon.setAttribute('spellcheck','false')
7058   gjwin.appendChild(gjicon)
7059
7060   gjtext = GJLog_Text = document.createElement('textarea')
7061   gjtext.addEventListener('keydown',GJWin_OnKeyDown)
7062   gjtext.addEventListener('keyup',GJWin_OnKeyUp)
7063   gjtext.addEventListener('resize',GJWin_OnResizeTextarea)
7064   gjtext.id = wid + '_Text'
7065   gjtext.setAttribute('class','GJText')
7066   gjtext.setAttribute('spellcheck','false')
7067   gjwin.appendChild(gjtext)
7068

```

```

7069
7070 // user's mode as of IME
7071 gjmode = GJWin_Mode = document.createElement('textarea')
7072 gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7073 gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7074 gjmode.id = wid + '_Mode'
7075 gjmode.setAttribute('class','GJMode')
7076 gjmode.setAttribute('spellcheck','false')
7077 gjmode.innerHTML = '---'
7078 gjwin.appendChild(gjmode)
7079
7080 gjwin.zIndex = 30000
7081 GJFactory.appendChild(gjwin)
7082
7083 gjtab.scrollTop = 0
7084 gjstat.scrollTop = 0
7085
7086 //x = gjwin.getBoundingClientRect().left.toFixed(0)
7087 //y = gjwin.getBoundingClientRect().top.toFixed(0)
7088 //gjwin.style.position = 'static'
7089 //gjwin.style.left = 0
7090 //gjwin.style.top = 0
7091
7092 //update = {'+wid+'.value=DateShort()});
7093 update = 'GJ_showTime1('+wid+')';
7094 // 2020-09-19 this causes memory leaks
7095 //FProductInterval = window.setInterval(update,200)
7096 //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
7097 //FProductInterval = window.setInterval(GJ_showTime1,200,wid);
7098 FProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
7099 return update
7100 }
7101 function xxxGJF_StripClass(){
7102 GJLog_Win.style.removeProperty('width')
7103 GJLog_Tab.style.removeProperty('width')
7104 GJLog_Stat.style.removeProperty('width')
7105 GJLog_Text.style.removeProperty('width')
7106 return "Stripped classes"
7107 }
7108 function isElem(id){
7109 return document.getElementById(id) != null
7110 }
7111 function GJLog_append(...args){
7112 txt = GJLog_Text;
7113 if( txt == null ){
7114 return; // maybe GJLog element is removed
7115 }
7116 logs = args.join(' ')
7117 txt.value += logs + '\n'
7118 txt.scrollTop = txt.scrollHeight
7119 //GJLog_Stat.value = DateShort()
7120 }
7121 //window.addEventListener('time',GJLog_StatUpdate)
7122 function test_GJ_Console(){
7123 window.setInterval(GJLog_StatUpdate,1000);
7124 GJ_NewConsole('GJ_Console')
7125 e = GJFactory;
7126 console.log('GJF0 #'+e.id+' from w='+e.style.width+', h='+e.style.height)
7127 e.style.width = 360; //GJFsize
7128 e.style.height = 320;
7129 console.log("GJF0 #"+e.id+" to w='"+e.style.width+', h='"+e.style.height)
7130 }
7131 /// test_GJ_Console();
7132
7133 var StopConsoleLog = true
7134 // 2020-09-15 added,
7135 // log should be saved to permanent memory
7136 // const px = new Proxy(console.log,{ alert() })
7137 __console_log = console.log
7138 __console_info = console.info
7139 __console_warn = console.warn
7140 __console_error = console.error
7141 __console_exception = console.exception
7142 // should pop callstack info.
7143 __console_exception = function(...args){
7144 __console_exception(...args)
7145 alert('-- got console.exception("'+args+'")')
7146 }
7147 __console_error = function(...args){
7148 __console_error(...args)
7149 alert('-- got console.error("'+args+'")')
7150 }
7151 __console_warn = function(...args){
7152 __console_warn(...args)
7153 alert('-- got console.warn("'+args+'")')
7154 }
7155 __console_info = function(...args){
7156 alert('-- got console.info("'+args+'")')
7157 __console_info(...args)
7158 }
7159 __console_xxxlog = function(...args){ // rewrite xxxlog to log to enable it
7160 __console_log(...args)
7161 if( StopConsoleLog ){
7162 return;
7163 }
7164 if( 0 <= args[0].indexOf('!') ){
7165 //alert('-- got console.log("'+args+'")')
7166 }
7167 GJLog_append(...args)
7168 }
7169
7170 //document.getElementById('GshFaviconURL').href = GShellFavicon
7171 //document.getElementById('GshFaviconURL').href = GshellInsideIcon
7172 //document.getElementById('GshFaviconURL').href = ITSmoreQR
7173 //document.getElementById('GshFaviconURL').href = GSellLogo
7174
7175 // id of GShell HTML elements
7176 var E_BANNER = "GshBanner" // banner element in HTML
7177 var E_FOOTER = "GshFooter" // footer element in HTML
7178 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7179 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7180 var E_TODO = "gsh-todo" // TODO of GShell
7181 var E_DICT = "gsh-dict" // Dictionary of GShell
7182
7183 function bannerElem(){ return document.getElementById(E_BANNER); }
7184 function bannerStyleFunc(){ return bannerElem().style; }
7185 var bannerStyle = bannerStyleFunc()
7186 function GshSetIcons(){
7187 document.getElementById('GshFaviconURL').href = GShellInsideIcon
7188 bannerStyle.backgroundImage = "url(\"+GsellLogo+)";
7189 //bannerStyle.backgroundImage = "url(\"+GshellInsideIcon+)";
7190 //bannerStyle.backgroundImage = "url(\"+GshellFavicon+)";
7191 GMenu.style.backgroundImage = "url(\"+GshellInsideIcon+)";
7192 showFooter();

```

```

7193 }
7194
7195 function footerElem(){ return document.getElementById(E_FOOTER); }
7196 function footerStyle(){ return footerElem().style; }
7197 //footerElem().style.backgroundImage="url(\"+ITSmoreQR+\")",
7198 //footerStyle().backgroundImage = "url(\"+ITSmoreQR+\")";
7199
7200 function html_fold(e){
7201   if( e.innerHTML == "Fold" ){
7202     e.innerHTML = "Unfold"
7203     document.getElementById('gsh-menu-exit').innerHTML=""
7204     document.getElementById('GshStatement').open=false
7205     GshFeatures.open = false
7206     document.getElementById('html-src').open=false
7207     document.getElementById(E_GINDEX).open=false
7208     document.getElementById(E_GOCODE).open=false
7209     document.getElementById(E_TODO).open=false
7210     document.getElementById('references').open=false
7211   }else{
7212     e.innerHTML = "Fold"
7213     document.getElementById('GshStatement').open=true
7214     GshFeatures.open = true
7215     document.getElementById(E_GINDEX).open=true
7216     document.getElementById(E_GOCODE).open=true
7217     document.getElementById(E_TODO).open=true
7218     document.getElementById('references').open=true
7219   }
7220 }
7221 function html_pure(e){
7222   if( e.innerHTML == "Pure" ){
7223     document.getElementById('gsh').style.display=true
7224     //document.style.display = false
7225     e.innerHTML = "Unpure"
7226   }else{
7227     document.getElementById('gsh').style.display=false
7228     //document.style.display = true
7229     e.innerHTML = "Pure"
7230   }
7231 }
7232
7233 var bannerIsStopping = false
7234 //NOTE: .com/JSREF/prop_style_backgroundposition.asp
7235 function shiftBG(){
7236   bannerIsStopping = !bannerIsStopping
7237   bannerStyle.backgroundPosition = "0 0";
7238 }
7239 // status should be inherited on Window Fork(), so use the status in DOM
7240 function html_stop(e,toggle){
7241   if( toggle ){
7242     if( e.innerHTML == "Stop" ){
7243       bannerIsStopping = true
7244       e.innerHTML = "Start"
7245     }else{
7246       bannerIsStopping = false
7247       e.innerHTML = "Stop"
7248     }
7249   }else{
7250     // update JavaScript variable from DOM status
7251     if( e.innerHTML == "Stop" ){ // shown if it's running
7252       bannerIsStopping = false
7253     }else{
7254       bannerIsStopping = true
7255     }
7256   }
7257 }
7258 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
7259 //html_stop(bannerElem(),false) // onInit.
7260
7261 //https://www.w3schools.com/jsref/met_win_setinterval.asp
7262 function shiftBanner(){
7263   var now = new Date().getTime();
7264   //console.log("now"+(now%10))
7265   if( !bannerIsStopping ){
7266     bannerStyle.backgroundPosition = ((now/10)%100000)+" 0";
7267   }
7268 }
7269 function Banner_init(){
7270   window.setInterval(shiftBanner,10); // onInit.
7271 }
7272
7273 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
7274 // from embedded html to standalone page
7275 var MyChildren = 0
7276 function html_fork(){
7277   GJFactory_Destroy()
7278   MyChildren += 1
7279   WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
7280   newwin = window.open("",WinId,"");
7281   src = document.getElementById("gsh");
7282   srchtml = src.outerHTML
7283   newwin.document.write("/*<"+"html>\n");
7284   newwin.document.write(srchtml);
7285   newwin.document.write("<"+"/"+"html>\n");
7286   newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
7287   newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
7288   newwin.document.close();
7289   newwin.focus();
7290 }
7291 function html_close(){
7292   window.close()
7293 }
7294 function win_jump(win){
7295   //win = window.top;
7296   win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
7297   if( win == null ){
7298     console.log("jump to window.opener(\"+win+)\")\n")
7299   }else{
7300     console.log("jump to window.opener(\"+win+)\")\n")
7301     win.focus();
7302   }
7303 }
7304
7305 // 0.2.9 2020-0902 created checksum of HTML
7306 CRC32UNIX = 0x04C1DB7 // Unix cksum
7307 function byteCRC32add(bigcrc,octstr,octlen){
7308   var crc = new Int32Array(1)
7309   crc[0] = bigcrc
7310
7311   let oi = 0
7312   for( ; oi < octlen; oi++ ){
7313     var oct = new Int8Array(1)
7314     oct[0] = octstr[oi]
7315     for( bi = 0; bi < 8; bi++ ){
7316       //console.log("-CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+"."+bi+"]\n")
    
```

```

7317         ovf1 = crc[0] < 0 ? 1 : 0
7318         ovf2 = oct[0] < 0 ? 1 : 0
7319         ovf = ovf1 ^ ovf2
7320         oct[0] <= 1
7321         crc[0] <= 1
7322         if( ovf ){ crc[0] ^= CRC32UNIX }
7323     }
7324 }
7325 //console.log("--CRC32 byteAdd return crc="+crc[0]+", "+oi+"/"+octlen+"\n")
7326 return crc[0];
7327 }
7328 function strCRC32add(bigcrc,str,strlen){
7329     var crc = new Uint32Array(1)
7330     crc[0] = bigcrc
7331     var code = new Uint8Array(strlen);
7332     for( i = 0; i < strlen; i++){
7333         code[i] = stri.charCodeAt(i) // not charAt() !!!!  

7334         //console.log("== "+code[i].toString(16)+" <== "+stri[i]+\n")
7335     }
7336     crc[0] = byteCRC32add(crc,code,strlen)
7337 //console.log("--CRC32 strAdd return crc="+crc[0]+\n")
7338     return crc[0]
7339 }
7340 function byteCRC32end(biggcrc,len){
7341     var crc = new Uint32Array(1)
7342     crc[0] = biggcrc
7343     var selen = new Uint8Array(4)
7344     let li = 0
7345     for( ; li < 4; ){
7346         selen[li] = len
7347         li += 1
7348         len >= 8
7349         if( len == 0 ){
7350             break
7351         }
7352     }
7353     crc[0] = byteCRC32add(crc,selen,li)
7354     crc[0] ^= 0xFFFFFFFF
7355     return crc[0]
7356 }
7357 function strCRC32(stri,len){
7358     var crc = new Uint32Array(1)
7359     crc[0] = 0
7360     crc[0] = strCRC32add(0,stri,len)
7361     crc[0] = byteCRC32end(crc[0],len)
7362 //console.log("--CRC32 "+crc[0]+" "+len+"\n")
7363     return crc[0]
7364 }
7365
7366 DestroyGJLink = null; // to be replaced
7367 DestroyFooter = null; // to be defined
7368 DestroyEventSharingCodeview = function dummy(){}
7369 Destroy_WirtualDesktop = function(){}
7370 DestroyNaviButtons = function(){}
7371
7372 function getSourceText(){
7373     if( DestroyFooter != null ) DestroyFooter();
7374     version = document.getElementById('GshVersion').innerHTML
7375     sfavico = document.getElementById('GshFaviconURL').href;
7376     sbanner = document.getElementById('GshBanner').style.backgroundImage;
7377     spositi = document.getElementById('GshBanner').style.backgroundPosition;
7378
7379     if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
7380     if( DestroyGJLink != null ) DestroyGJLink();
7381     DestroyEventSharingCodeview();
7382     Destroy_WirtualDesktop();
7383     GshTopbar.innerHTML = "";
7384     DestroyIndexBar();
7385     DestroyNaviButtons();
7386
7387     // these should be removed by CSS selector or class, after seavaed to non-printed attribute
7388     GshBanner.removeAttribute('style');
7389     document.getElementById('GshMenuSign').removeAttribute("style");
7390     styleGMenu = GMenu.getAttribute("style")
7391     GMENU.removeAttribute("style");
7392     styleGStat = GStat.getAttribute("style")
7393     GStat.removeAttribute("style");
7394     styleGTop = GTop.getAttribute("style")
7395     GTOP.removeAttribute("style");
7396     styleGshGrid = GshGrid.getAttribute("style")
7397     GshGrid.removeAttribute("style");
7398     //styleGPos = GPos.getAttribute("style");
7399     //GPos.removeAttribute("style");
7400     //GPos.innerHTML = "";
7401     //styleGLog = GLog.getAttribute("style");
7402     //GLog.removeAttribute("style");
7403     //GLog.innerHTML = "";
7404     styleGshellPlane = GshellPlane.getAttribute("style")
7405     GshellPlane.removeAttribute("style")
7406     styleRawTextViewer = RawTextViewer.getAttribute("style")
7407     RawTextViewer.removeAttribute("style")
7408     styleRawTextViewerClose = RawTextViewerClose.getAttribute("style")
7409     RawTextViewerClose.removeAttribute("style")
7410
7411     GshFaviconURL.href = "";
7412     if( iselem('ConfigIcon') ) ConfigIcon.src = "";
7413
7414     //it seems that interHTML and outerHTML generate style="" for these (??)
7415     //GshBanner.removeAttribute('style');
7416     //GshFooter.removeAttribute('style');
7417     //GshMenuSign.removeAttribute('style');
7418     GshBanner.style="";
7419     GshMenuSign.style="";
7420
7421     textarea = document.createElement("textarea")
7422     srchtml = document.getElementById("gsh").outerHTML;
7423     //textarea = document.createElement("textarea")
7424     // 2020-0910 ?? ... this causes inserting style="" to Banner and Footer,
7425     // with Chromium?/ after reloading from file:///
7426     textarea.innerHTML = srchtml
7427     // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7428     var rawtext = textarea.value
7429     //textarea.destroy()
7430     //rawtext = gsh.textContent // this removes #include <FILENAME> too
7431     var orgtext = ""
7432     + /*<*/"html>\n" // lost preamble text
7433     + rawtext
7434     + "<"/"html>\n" // lost trail text
7435     ;
7436
7437     tlen = orgtext.length
7438     //console.log("getSourceText: length="+tlen+"\n")
7439     document.getElementById('GshFaviconURL').href = sfavico;
7440

```

```

7441 document.getElementById('GshBanner').style.backgroundImage    = sbanner;
7442 document.getElementById('GshBanner').style.backgroundPosition = spositi;
7443
7444 GStat.setAttribute("style",styleGStat)
7445 GMenu.setAttribute("style",styleGMenu)
7446 GTop.setAttribute("style",styleGTop)
7447 //GLog.setAttribute("style",styleGLog)
7448 //GPos.setAttribute("style",styleGPos)
7449 GshGrid.setAttribute("style",styleGshGrid)
7450 GShellPlane.setAttribute("style",styleGshellPlane)
7451 RawTextViewer.setAttribute("style",styleRawTextViewer)
7452 RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
7453 canontext = orgtext.replace(' style=""','');
7454 // open="" too
7455 return canontext
7456 }
7457 function getDigest(){
7458     var text = "";
7459     text = getSourceText()
7460     var digest = ""
7461     tlen = text.length
7462     digest = strCRC32(text,tlen) + " " + tlen
7463     return { text, digest }
7464 }
7465 function html_digest(){
7466     version = document.getElementById('GshVersion').innerHTML
7467     let {text, digest} = getDigest()
7468     alert("cksum: " + digest + " " + version)
7469 }
7470 function charsin(stri,char){
7471     ln = 0;
7472     for( i = 0; i < stri.length; i++ ){
7473         if( stri.charCodeAt(i) == char.charCodeAt(0) )
7474             ln++;
7475     }
7476     return ln;
7477 }
7478
7479 //<script>customElements.define('digest',digestElement)</script>
7480
7481 function showDigest(e){
7482     result = 'version=' + GshVersion.innerHTML + '\n'
7483     result += 'lines=' + e.dataset.lines + '\n'
7484     result += 'length=' + e.dataset.length + '\n'
7485     result += 'crc32u=' + e.dataset.crc32u + '\n'
7486     result += 'time=' + e.dataset.time + '\n';
7487
7488     alert(result)
7489 }
7490
7491 function html_sign(e){
7492     if( RawTextViewer.style.zIndex == 1000 ){
7493         hideRawTextViewer()
7494         return
7495     }
7496     GshTopbar.innerHTML = "";
7497     DestroyIndexBar();
7498     DestroyNavButtons();
7499     DestroyEventSharingCodeview();
7500     Destroy_Virtualdesktop();
7501     GJFactory_Destroy()
7502     if( DestroyGJLink != null ) DestroyGJLink();
7503     //gsh_digest.innerHTML = "";
7504     text = getSourceText() // the original text
7505     tlen = text.length
7506     digest = strCRC32(text,tlen)
7507     //gsh_digest.innerHTML = digest + " " + tlen
7508     //text = getSourceText() // the text with its digest
7509     Lines = charsin(text,'\n')
7510
7511     name = "gsh"
7512     sid = name + "-digest"
7513     d = new Date()
7514     signedAt = d.getTime()
7515
7516     sign = '/'+*<'+span\n'
7517     sign += ' id="' + sid + '\n'
7518     sign += ' class="digest"\n'
7519     sign += ' data-target-id="'+name+'\n'
7520     sign += ' data-crc32u=' + digest + '\n'
7521     sign += ' data-length=' + tlen + '\n'
7522     sign += ' data-lines=' + Lines + '\n'
7523     sign += ' data-time=' + signedAt + '\n'
7524     sign += '><' + '/span>\n*'+'\n'
7525
7526     text = sign + text
7527
7528     txthtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td>'
7529     txthtml += '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">'
7530     for( i = 1; i <= Lines; i++ ){
7531         txthtml += i.toString() + '\n'
7532     }
7533     txthtml += "
7534     + '<' + '/textarea>'
7535     + '<' + '/td><' + 'td>',
7536     + '<' + 'textarea cols=150 rows=' + Lines + ' spellcheck="false"'
7537     + ' class="LineNumbered">'
7538     + text + '<' + '/textarea>'
7539     + '<' + '/td><' + '/tr><' + '/table>';
7540
7541     for( i = 1; i <= 30; i++ ){
7542         txthml += '<br>\n'
7543     }
7544     RawTextViewer.innerHTML = txthml
7545     RawTextViewer.spellcheck = false // (spellcheck above seems ineffective)
7546
7547     btn = e
7548     e.style.color = "rgba(128,128,255,0.9)";
7549     y = e.getBoundingClientRect().top.toFixed(0)
7550     //h = e.getBoundingClientRect().height.toFixed(0)
7551     RawTextViewer.style.top = Number(y) + 30
7552     RawTextViewer.style.left = 100;
7553     RawTextViewer.style.height = window.innerHeight - 20;
7554     //RawTextViewer.style.opacity = 1.0;
7555     RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
7556     RawTextViewer.style.background-color = "rgba(255,255,255,0.8)";
7557     RawTextViewer.style.zIndex = 1000;
7558     RawTextViewer.style.display = true;
7559
7560     if( RawTextViewerClose.style == null ){
7561         RawTextViewerClose.style = "";
7562     }
7563     RawTextViewerClose.style.top = Number(y) + 10
7564     RawTextViewerClose.style.left = 100;

```

```

7565     RawTextViewerClose.style.zIndex = 1001;
7566
7567     ScrollToElement(CurElement,RawTextViewerClose)
7568 }
7569 function hideRawTextViewer(){
7570     RawTextViewer.style.left = 10000;
7571     RawTextViewer.style.zIndex = -100;
7572     RawTextViewer.style.opacity = 0.0;
7573     RawTextViewer.style = null
7574     RawTextViewer.innerHTML = "";
7575
7576     GshMenuSign.style.color = "rgba(255,128,128,1.0)";
7577     RawTextViewerClose.style.top = 0;
7578     RawTextViewerClose.style = null
7579 }
7580
7581 // source code viewer
7582 function frame_close(){
7583     srcframe = document.getElementById("src-frame");
7584     srcframe.innerHTML = "";
7585     //srcframe.style.cols = 1;
7586     srcframe.style.rows = 1;
7587     srcframe.style.height = 0;
7588     srcframe.style.display = false;
7589     src = document.getElementById("SrcTextarea");
7590     src.innerHTML = ""
7591     //src.cols = 0
7592     src.rows = 0
7593     src.display = false
7594     //alert("--closed--")
7595 }
7596 //<!-- | <span onclick="html_view();">Source</span> -->
7597 //<!-- | <span onclick="frame_close();">SourceClose</span> -->
7598 //<!--| <span>Download</span> -->
7599 function frame_open(){
7600     GshTopbar.innerHTML = "";
7601     DestroyIndexBar();
7602     DestroyNavButtons();
7603     if( DestroyFooter != null ) DestroyFooter();
7604     document.getElementById('GshFaviconURL').href = "";
7605     oldsrc = document.getElementById("GENSRC");
7606     if( oldsrc != null ){
7607         //alert("-I--(erasing old text)")
7608         oldsrc.innerHTML = "";
7609         return
7610     }else{
7611         //alert("--I--(no old text)")
7612     }
7613     styleBanner = GshBanner.getAttribute("style")
7614     GshBanner.removeAttribute("style")
7615     if( document.getElementById('GJC_1').remove() ) { GJC_1.remove() }
7616
7617     GshFaviconURL.href = "";
7618     if( iselem('ConfigIcon') ) ConfigIcon.src = "";
7619     GStat.removeAttribute('style')
7620     GshGrid.removeAttribute('style')
7621     GshMenuSign.removeAttribute('style')
7622     //GPos.removeAttribute('style')
7623     //GPos.innerHTML = "";
7624     //GLog.removeAttribute('style')
7625     //GLog.innerHTML = "";
7626     GMenu.removeAttribute('style')
7627     GTop.removeAttribute('style')
7628     GShellPlane.removeAttribute('style')
7629     RawTextViewer.removeAttribute('style')
7630     RawTextViewerClose.removeAttribute('style')
7631
7632     if( DestroyGJLink != null ) DestroyGJLink();
7633     GJFactory_Destroy()
7634     Destroy_WirtualDesktop();
7635     DestroyEventSharingCodeview();
7636
7637     src = document.getElementById("gsh");
7638     srchtml = src.outerHTML
7639     srcframe = document.getElementById("src-frame");
7640     srcframe.innerHTML = ""
7641     + "<+"+cite id='GENSRC'>\n"
7642     + "<+"+style+"\n"
7643     + "#GENSRC textarea{tab-size:4;}\n"
7644     + "#GENSRC textarea{-o-tab-size:4;}\n"
7645     + "#GENSRC textarea{-moz-tab-size:4;}\n"
7646     + "#GENSRC textarea{spellcheck:false;}\n"
7647     + "</"+"style>\n"
7648     + "<+"+textarea id='SrcTextarea' cols=100 rows=20 class="gsh-code" spellcheck="false">\n"
7649     + "</"+"html>\n" // lost preamble text
7650     + srchtml
7651     + "<+"+"/html>\n" // lost trail text
7652     + "<+"+textarea>\n"
7653     + "</"+"cite><!-- GENSRC -->\n";
7654
7655     //srcframe.style.cols = 80;
7656     //srcframe.style.rows = 80;
7657
7658     GshBanner.setAttribute('style',styleBanner)
7659 }
7660 function fill_CSSview(){
7661     part = document.getElementById('GshStyleDef')
7662     view = document.getElementById('gsh-style-view')
7663     view.innerHTML = ""
7664     + "<+"+textarea cols=100 rows=20 class="gsh-code">\n"
7665     + part.innerHTML
7666     + "<+"+"/textarea>\n"
7667 }
7668 function fill_JavaScriptView(){
7669     jspart = document.getElementById('gsh-script')
7670     view = document.getElementById('gsh-script-view')
7671     view.innerHTML = ""
7672     + "<+"+textarea cols=100 rows=20 class="gsh-code">\n"
7673     + jspart.innerHTML
7674     + "<+"+"/textarea>\n"
7675 }
7676 function fill_DataView(){
7677     part = document.getElementById('gsh-data')
7678     view = document.getElementById('gsh-data-view')
7679     view.innerHTML = ""
7680     + "<+"+textarea cols=100 rows=20 class="gsh-code">\n"
7681     + part.innerHTML
7682     + "<+"+"/textarea>\n"
7683 }
7684 function jump_to_StyleView(){
7685     jsview = document.getElementById('html-src')
7686     jsview.open = true
7687     jsview = document.getElementById('gsh-style-frame')
7688     jsview.open = true

```

```

7689     fill_CSSview()
7690   }
7691   function jumpTo_JavaScriptView(){
7692     jsview = document.getElementById('html-src')
7693     jsview.open = true
7694     jsview = document.getElementById('gsh-script-frame')
7695     jsview.open = true
7696     fill_JavaScriptView()
7697   }
7698   function jumpTo_DataView(){
7699     jsview = document.getElementById('html-src')
7700     jsview.open = true
7701     jsview = document.getElementById('gsh-data-frame')
7702     jsview.open = true
7703     fill_DataView()
7704   }
7705   function jumpTo_WholeView(){
7706     jsview = document.getElementById('html-src')
7707     jsview.open = true
7708     jsview = document.getElementById('gsh-whole-view')
7709     jsview.open = true
7710     frame_open()
7711   }
7712   function html_view(){
7713     html_stop();
7714
7715     banner = document.getElementById('GshBanner').style.backgroundImage;
7716     footer = document.getElementById('GshFooter').style.backgroundImage;
7717     document.getElementById('GshBanner').style.backgroundImage = "";
7718     document.getElementById('GshBanner').style.backgroundPosition = "";
7719     document.getElementById('GshFooter').style.backgroundImage = "";
7720
7721 //srcwin = window.open("", "CodeView2", "");
7722 //srcwin = window.open("", "", "");
7723 srcwin.document.write("<span id=\"gsh\">\n");
7724
7725 src = document.getElementById("gsh");
7726 srcwin.document.write("<"+style>\n");
7727 srcwin.document.write("textarea{tab-size:4;}\n");
7728 srcwin.document.write("textareao{tab-size:4;}\n");
7729 srcwin.document.write("textareamoz{tab-size:4;}\n");
7730 srcwin.document.write("</style>\n");
7731 srcwin.document.write("<h2>\n");
7732 srcwin.document.write("<"+span onclick="window.close();\">Close</span> | \n");
7733 //srcwin.document.write("<"+span onclick="\"html_stop();\">Run</span>\n");
7734 srcwin.document.write("</h2>\n");
7735 srcwin.document.write("<"+textarea id="gsh-src-src" cols=100 rows=60>");
7736 srcwin.document.write("/<"+html>\n");
7737 srcwin.document.write("<"+span id="gsh">");
7738 srcwin.document.write(src.innerHTML);
7739 srcwin.document.write("<"+span><"+/html>\n");
7740 srcwin.document.write("</"+"textarea>\n");
7741
7742 document.getElementById('GshBanner').style.backgroundImage = banner;
7743 document.getElementById('GshFooter').style.backgroundImage = footer
7744
7745 sty = document.getElementById("GshStyleDef");
7746 srcwin.document.write("<"+style>\n");
7747 srcwin.document.write(sty.innerHTML);
7748 srcwin.document.write("<"+style>\n");
7749
7750 run = document.getElementById("gsh-script");
7751 srcwin.document.write("<"+script>\n");
7752 srcwin.document.write(run.innerHTML);
7753 srcwin.document.write("<"+script>\n");
7754
7755 srcwin.document.write("<"+span><"+/html>\n"); // gsh span
7756 srcwin.document.close();
7757 srcwin.focus();
7758 }
7759 GSH = document.getElementById("gsh")
7760
7761 //GSH.onclick = "alert('Ouch!')"
7762 //GSH.css = "{background-color:#eef;}"
7763 //GSH.style = "background-color:#eef;";
7764 //GSH.style.display = false;
7765 //alert('Ouch01')
7766 //GSH.style.display = true;
7767
7768 // 2020-0904 created, tentative
7769 document.addEventListener('keydown', jgshCommand);
7770 //CurElement = GshStatement
7771 CurElement = GshMenu
7772 MemElement = GshMenu
7773
7774 function nextSib(e){
7775   n = e.nextSibling;
7776   for( i = 0; i < 100; i++ ){
7777     if( n == null ){
7778       break;
7779     }
7780     if( n.nodeName == "DETAILS" ){
7781       return n;
7782     }
7783     n = n.nextSibling;
7784   }
7785   return null;
7786 }
7787 function prevSib(e){
7788   n = e.previousSibling;
7789   for( i = 0; i < 100; i++ ){
7790     if( n == null ){
7791       break;
7792     }
7793     if( n.nodeName == "DETAILS" ){
7794       return n;
7795     }
7796     n = n.previousSibling;
7797   }
7798   return null;
7799 }
7800 function setColor(e,eName,eColor){
7801   if( e.hasChildNodes() ){
7802     s = e.childNodes;
7803     if( s != null ){
7804       for( ci = 0; ci < s.length; ci++ ){
7805         if( s[ci].nodeName == eName ){
7806           s[ci].style.color = eColor;
7807           //s[ci].style.backgroundColor = eColor;
7808           break;
7809         }
7810       }
7811     }
7812   }
}

```

```

7813 }
7814
7815 // https://docs.microsoft.com/en-us/previous-versions/hh781509(v=vs.85)
7816 function showCurElementPosition(ev){
7817 // if( document.getElementById("GPos") == null ){
7818 //   return;
7819 //
7820 //   if( GPos == null ){
7821 //     return;
7822 //
7823   e = CurElement
7824   y = e.getBoundingClientRect().top.toFixed(0)
7825   x = e.getBoundingClientRect().left.toFixed(0)
7826
7827   h = ev + " "
7828   h += "y='"+y+"', "+ 'x='+'x+" -- "
7829   h += "w='"+ window.innerWidth + ", h='"+ window.innerHeight + " -- "
7830 //GPos.test = h
7831 //GPos.innerHTML = h
7832 // GPos.innerHTML = h
7833 }
7834
7835 function zero2(n){
7836   if( n < 10 ){
7837     return '0' + n;
7838   }else{
7839     return n;
7840   }
7841 }
7842 function DateHourMin(){
7843   d = new Date();
7844   //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes())
7845   return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
7846 }
7847 function DateShort0(d){
7848   return d.getFullYear()
7849   + ' / ' + zero2(d.getMonth())
7850   + ' / ' + zero2(d.getDate())
7851   + ' ' + zero2(d.getHours())
7852   + ':' + zero2(d.getMinutes())
7853   + ':' + zero2(d.getSeconds())
7854 }
7855 function DateShort(){
7856   return DateShort0(new Date());
7857 }
7858 function DateLong0(ms){
7859   d = new Date();
7860   d.setTime(ms);
7861   return DateShort0(d)
7862   + '.' + d.getMilliseconds()
7863   + ' ' + d.getTimezoneOffset()/60
7864   + ' ' + d.getTime() + '.' + d.getMilliseconds()
7865 }
7866 function DateLong(){
7867   return DateLong0(new Date());
7868 }
7869 function GShellMenu(e){
7870   //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
7871   showGShellPlane()
7872 }
7873 // placements of planes
7874 function GShellResizeX(ev){
7875   //if( document.getElementById("GMenu") != null ){
7876   //  GMenu.style.left = window.innerWidth - 100
7877   //  GMenu.style.top = window.innerHeight - 90 - 200
7878   //  //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)
7879
7880   //}
7881 GStat.style.width = window.innerWidth
7882 //if( document.getElementById("GPos") != null ){
7883 //  GPos.style.width = window.innerWidth
7884 //  GPos.style.top = window.innerHeight - 30; //GPos.style.height
7885 //}
7886 //if( document.getElementById("GLog") != null ){
7887 //  GLog.style.width = window.innerWidth
7888 //  GLog.innerHTML = ""
7889 //}
7890 //if( document.getElementById("GLog") != null ){
7891 //  GLog.innerHTML = "Resize: w=" + window.innerWidth +
7892 //  //, h=" + window.innerHeight
7893 //}
7894 showCurElementPosition(ev)
7895 }
7896 function GShellResize(){
7897   GShellResizeX("[RESIZE]")
7898 }
7899 window.onresize = GShellResize
7900 var prevNode = null
7901 var LogMouseMoveOverElement = false;
7902 function GJSH_OnMouseMove(ev){
7903   if( LogMouseMoveOverElement == false ){
7904     return;
7905   }
7906   x = ev.clientX
7907   y = ev.clientY
7908   d = new Date()
7909   t = d.getTime() / 1000
7910   if( document.elementFromPoint ){
7911     e = document.elementFromPoint(x,y)
7912     if( e != null ){
7913       if( e == prevNode ){
7914         else{
7915           console.log('Mo-' +t+ '('+x+', '+y+') '
7916           +e.nodeType+ ' '+e.tagName+ '#' +e.id)
7917           prevNode = e
7918         }
7919       }else{
7920         console.log(t+ '('+x+', '+y+') no element')
7921       }
7922     }else{
7923       console.log(t+ '('+x+', '+y+') no elementFromPoint')
7924     }
7925 }
7926 window.addEventListener('mousemove',GJSH_OnMouseMove);
7927
7928 function GJSH_OnMouseMoveScreen(ev){
7929   x = ev.screenX
7930   y = ev.screenY
7931   d = new Date()
7932   t = d.getTime() / 1000
7933   console.log(t+ '('+x+', '+y+') no elementFromPoint')
7934 }
7935 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
7936

```

```

7937 function ScrollToElement(oe,ne){
7938     ne.scrollIntoView()
7939     ny = ne.getBoundingClientRect().top.toFixed(0)
7940     nx = ne.getBoundingClientRect().left.toFixed(0)
7941     //GLog.innerHTML = "["+ny+","+nx+"]"
7942     //window.scrollTo(0,0)
7943
7944     GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
7945     GshGrid.style.left = '250px';
7946     GshGrid.style.zIndex = 0
7947     if( false ){
7948         oy = oe.getBoundingClientRect().top.toFixed(0)
7949         ox = oe.getBoundingClientRect().left.toFixed(0)
7950         y = e.getBoundingClientRect().top.toFixed(0)
7951         x = e.getBoundingClientRect().left.toFixed(0)
7952         window.scrollTo(x,y)
7953         ny = e.getBoundingClientRect().top.toFixed(0)
7954         nx = e.getBoundingClientRect().left.toFixed(0)
7955         //GLog.innerHTML = "("+oy+","+ox+")->("+y+","+x+")->["+ny+","+nx+"]"
7956     }
7957 }
7958 function showGShellPlane(){
7959     if( GShellPlane.style.zIndex == 0 ){
7960         GShellPlane.style.zIndex = 1000;
7961         GShellPlane.style.left = 30;
7962         GShellPlane.style.height = 320;
7963         GShellPlane.innerHTML = DateLong() + "<br>" +
7964             "-- History --<br>" + MyHistory;
7965     }else{
7966         GShellPlane.style.zIndex = 0;
7967         GShellPlane.style.left = 0;
7968         GShellPlane.style.height = 50;
7969         GShellPlane.innerHTML = "";
7970     }
7971 }
7972 var SuppressGJShell = false
7973 function jshCommand(keyevent){
7974     if( SuppressGJShell ){
7975         return
7976     }
7977     key = keyevent
7978     keycode = key.code
7979     //GStat.style.width = window.innerWidth
7980     GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
7981
7982     console.log("JSGsh-Key:"+keycode+"(^~)/*")
7983     if( keycode == "Slash" ){
7984         console.log('('+'x+', '+'y+') ')
7985         e = document.elementFromPoint(x,y)
7986         console.log('('+'x+', '+'y+') '+e.nodeType+' '+e.tagName+'#'+e.id)
7987     }else
7988     if( keycode == "Digit0" ){ // fold side-bar
7989         // "zero page"
7990         showGShellPlane();
7991     }else
7992     if( keycode == "Digit1" ){ // fold side-bar
7993         primary.style.width = "94%"
7994         secondary.style.width = "0%"
7995         secondary.style.opacity = 0
7996         GStat.innerHTML = "[Single Column View]"
7997     }else
7998     if( keycode == "Digit2" ){ // unfold side-bar
7999         primary.style.width = "58%"
8000         secondary.style.width = "36%"
8001         secondary.style.opacity = 1
8002         GStat.innerHTML = "[Double Column View]"
8003     }else
8004     if( keycode == "KeyU" ){ // fold/unfold all
8005         html_fold(GshMenuFold);
8006         location.href = "#"+CurElement.id;
8007     }else
8008     if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8009         CurElement.open = !CurElement.open;
8010     }else
8011     if( keycode == "ArrowRight" ){ // unfold the element
8012         CurElement.open = true
8013     }else
8014     if( keycode == "ArrowLeft" ){ // unfold the element
8015         CurElement.open = false
8016     }else
8017     if( keycode == "KeyI" ){ // inspect the element
8018         e = CurElement
8019         //GLog.innerHTML =
8020         GJLog_append("Current Element: " + e + "<br>" +
8021             + "name='"+e.nodeName + "', "
8022             + "id='"+e.id + "', "
8023             + "children='"+e.childNodes.length + "', "
8024             + "parent='"+e.parentNode.id + "<br>" +
8025             + "text='"+e.textContent)
8026         GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8027         return
8028     }else
8029     if( keycode == "KeyM" ){ // memory the position
8030         MemElement = CurElement
8031     }else
8032     if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8033         e = nextSib(CurElement)
8034         if( e != null ){
8035             setColor(CurElement,"SUMMARY","#fff")
8036             setColor(e,"SUMMARY","#8f8") // should be complement ?
8037             oe = CurElement
8038             CurElement = e
8039             //location.href = "#"+e.id;
8040             ScrollToElement(oe,e)
8041         }
8042     }else
8043     if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8044         oe = CurElement
8045         e = prevSib(CurElement)
8046         if( e != null ){
8047             setColor(CurElement,"SUMMARY","#fff")
8048             setColor(e,"SUMMARY","#8f8") // should be complement ?
8049             CurElement = e
8050             //location.href = "#"+e.id;
8051             ScrollToElement(oe,e)
8052         }else{
8053             e = document.getElementById("GshBanner")
8054             if( e != null ){
8055                 setColor(CurElement,"SUMMARY","#fff")
8056                 CurElement = e
8057                 ScrollToElement(oe,e)
8058             }else{
8059                 e = document.getElementById("primary")
8060                 if( e != null ){

```

```

8061         setColor(CurElement,"SUMMARY","#fff")
8062         CurElement = e
8063         ScrollToElement(oe,e)
8064     }
8065   }
8066 }else
8067 if( keycode == "KeyR" ){
8068   location.reload()
8069 }else
8070 if( keycode == "KeyJ" ){
8071   GshGrid.style.top = '120px';
8072   GshGrid.innerHTML = '(>_<){Down}';
8073 }else
8074 if( keycode == "KeyK" ){
8075   GshGrid.style.top = '0px';
8076   GshGrid.innerHTML = '(^_-^){Up}';
8077 }else
8078 if( keycode == "KeyH" ){
8079   GshGrid.style.left = '0px';
8080   GshGrid.innerHTML = "(-_-){Left}";
8081 }else
8082 if( keycode == "KeyL" ){
8083   //GLog.innerHTML +=
8084   GJLog_append(
8085     'screen=' + screen.width + 'px' + '<br>' +
8086     'window=' + window.innerWidth + 'px' + '<br>' +
8087   )
8088   GshGrid.style.left = (document.documentElement.clientWidth - 160).toString(10) + 'px';
8089   GshGrid.innerHTML = '(@_@){Right}';
8090 }else
8091 if( keycode == "KeyS" ){
8092   html_stop(GshMenuStop, true)
8093 }else
8094 if( keycode == "KeyF" ){
8095   html_fork()
8096 }else
8097 if( keycode == "KeyC" ){
8098   window.close()
8099 }else
8100 if( keycode == "KeyD" ){
8101   html_digest()
8102 }else
8103 if( keycode == "KeyV" ){
8104   e = document.getElementById('gsh-digest')
8105   if( e != null ){
8106     showDigest(e)
8107   }
8108 }
8109 }
8110 showCurElementPosition("[+key.code+" --");
8111 //if( document.getElementById("GPos") != null ){
8112 //  GPos.innerHTML += "[+key.code+" --"
8113 //}
8114 //}
8115 //GShellResizeX("[+key.code+" --");
8116 }
8117 var initGSKC = false;
8118 function GShell_initKeyCommands(){
8119   if( initGSKC ) { return; } initGSKC = true;
8120
8121   GShellResizeX("[INIT]");
8122   DisplaySize = '-- Display:
8123     + 'screen=' + screen.width + 'px, +' + 'window=' + window.innerWidth + 'px';
8124
8125   let {text, digest} = getDigest()
8126   //GLog.innerHTML +=
8127   GJLog_append(
8128     '-- GShell: ' + GshVersion.innerHTML + '\n' +
8129     '-- Digest: ' + digest + '\n' +
8130     DisplaySize
8131     //+ "<br>" + "-- LastVisit:<br>" + MyHistory
8132   )
8133   GShellResizeX(null);
8134 }
8135 //GShell_initKeyCommands();
8136
8137 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8138 //Convert a string into an ArrayBuffer
8139 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8140 function str2ab(str) {
8141   const buf = new ArrayBuffer(str.length);
8142   const bufView = new Uint8Array(buf);
8143   for (let i = 0, strLen = str.length; i < strLen; i++) {
8144     bufView[i] = str.charCodeAt(i);
8145   }
8146   return buf;
8147 }
8148 function importPrivateKey(pem) {
8149   const binaryDerString = window.atob(pemContents);
8150   const binaryDer = str2ab(binaryDerString);
8151   return window.crypto.subtle.importKey(
8152     "pkcs8",
8153     binaryDer,
8154     {
8155       name: "RSA-PSS",
8156       modulusLength: 2048,
8157       publicExponent: new Uint8Array([1, 0, 1]),
8158       hash: "SHA-256",
8159     },
8160     true,
8161     ["sign"]
8162   );
8163 }
8164 //importPrivateKey(ppem)
8165
8166 //key = {}
8167 //buf = "abc"
8168 //enc = "xyxxxxxxxx"; //crypto.publicEncrypt(key,buf)
8169 //b64 = btoa(enc)
8170 //dec = atob(b64)
8171 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8172 </script>
8173 /*
8174 */
8175 /*
8176 <!-- ----- GJConsole BEGIN { ----- -->
8177 <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
8178 <details><summary>GJ Console</summary>
8179 <p>
8180 <span id="GJE_RootNode0"></span>
8181 <span id="GJC1_Container"></span>
8182 </p>
8183 <style id="GJConsoleStyle">
8184   .GJConsole {

```

```

8185     z-index:1000;
8186     width:400; height:200px;
8187     margin:2px;
8188     color:#fff; background-color:#66a;
8189     font-size:12px; font-family:monospace,Courier New;
8190   }
8191 </style>
8192
8193 <script id="GJConsoleScript" class="GJConsole">
8194   var PS1 = "% "
8195   function GJC_KeyDown(keyevent){
8196     key = keyevent.code
8197     if( key == "Enter" ){
8198       GJC_Command(this)
8199       this.value += "\n" + PS1 // prompt
8200     }else
8201     if( key == "Escape"){
8202       SuppressGJShell = false
8203       GshMenu.focus() // should be previous focus
8204     }
8205   }
8206   var GJC_SessionId
8207   function GJC_SetSessionId(){
8208     var xd = new Date()
8209     GJC_SessionId = xd.getTime() / 1000
8210   }
8211   GJC_SetSessionId()
8212   function GJC_Memory(mem,args,text){
8213     argv = args.split(' ')
8214     cmd = argv[0]
8215     argv.shift()
8216     args = argv.join(' ')
8217     ret = ""
8218
8219     if( cmd == 'clear' ){
8220       Permanent.setItem(mem,'')
8221     }else
8222     if( cmd == 'read' ){
8223       ret = Permanent.getItem(mem)
8224     }else
8225     if( cmd == 'save' ){
8226       val = Permanent.getItem(mem)
8227       if( val == null ) { val = "" }
8228       d = new Date()
8229       val += d.getTime()/1000+ " "+GJC_SessionId+ " "+document.URL+ " "+args+ "\n"
8230       val += text.value
8231       Permanent.setItem(mem,val)
8232     }else
8233     if( cmd == 'write' ){
8234       val = Permanent.getItem(mem)
8235       if( val == null ) { val = "" }
8236       d = new Date()
8237       val += d.getTime()/1000+ " "+GJC_SessionId+ " "+document.URL+ " "+args+ "\n"
8238       Permanent.setItem(mem,val)
8239     }else{
8240       ret = "Commands: write | read | save | clear"
8241     }
8242   return ret
8243 }
8244 // -- 2020-09-14 added TableEditor
8245 var GJE_CurElement = null; //GJE_RootNode
8246 GJE_NodeSaved = null
8247 GJE_TableNo = 1
8248 function GJE_StyleKeyCommand(kev){
8249   keycode = kev.code
8250   console.log('GJE-Key: '+keycode)
8251   if( keycode == 'Escape' ){
8252     GJE_SetStyle(this);
8253   }
8254   kev.stopPropagation()
8255   // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
8256 }
8257 var GJE_CommandMode = false
8258 function GJE_TableKeyCommand(kev,tab){
8259   wasCmdMode = GJE_CommandMode
8260   key = kev.code
8261   if( key == 'Escape' ){
8262     console.log("To command mode: "+tab.nodeName+"#"+tab.id)
8263     //tab.setAttribute('contenteditable','false')
8264     tab.style.caretColor = "blue"
8265     GJE_CommandMode = true
8266   }else
8267   if( key == "KeyA" ){
8268     tab.style.caretColor = "red"
8269     GJE_CommandMode = false
8270   }else
8271   if( key == "KeyL" ){
8272     tab.style.caretColor = "red"
8273     GJE_CommandMode = false
8274   }else
8275   if( key == "KeyO" ){
8276     tab.style.caretColor = "red"
8277     GJE_CommandMode = false
8278   }else
8279   if( key == "KeyJ" ){
8280     console.log("ROW-DOWN")
8281   }else
8282   if( key == "KeyK" ){
8283     console.log("ROW-UP")
8284   }else
8285   if( key == "KeyW" ){
8286     console.log("COL-FORW")
8287   }else
8288   if( key == "KeyB" ){
8289     console.log("COL-BACK")
8290   }
8291
8292   kev.stopPropagation()
8293   if( wasCmdMode ){
8294     kev.preventDefault()
8295   }
8296 }
8297 function GJE_DragEvent(ev,elem){
8298   x = ev.clientX
8299   y = ev.clientY
8300   console.log("dragged: "+this.nodeName+'#'+this.id+' x='+x+ ' y=' +y)
8301 }
8302 // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
8303 // https://www.w3.org/TR/uievents/#events-mouseevents
8304 function GJE_DropEvent(ev,elem){
8305   x = ev.clientX
8306   y = ev.clientY
8307   this.style.x = x
8308   this.style.y = y

```

```

8309     this.style.position = 'absolute' // 'fixed'
8310     this.parentNode = gsh // just for test
8311     console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+ ' y=' +y
8312         +' parent=' +this.parentNode.id)
8313   }
8314   function GJE_SetTableStyle(ev){
8315     this.innerHTML = this.value; // sync. for external representation?
8316     if(false){
8317       stid = this.parentNode.id+this.id
8318       // and remove "_span" at the end
8319       e = document.getElementById(stid)
8320       //alert('SetTableStyle #' + e.id + '\n' + this.value)
8321       if( e != null ){
8322         e.innerHTML = this.value
8323       }else{
8324         console.log('Style Not found: '+stid)
8325       }
8326       //alert('event StopPropagation: '+ev)
8327     }
8328   }
8329   function setCSofClass(cclass,cstyle){
8330     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
8331     rlen = ss.cssRules.length;
8332     let tabrule = null;
8333     rulex = -1
8334
8335     // should skip white space at the top of cstyle
8336     sel = cstyle.charAt(0);
8337     selector = sel+cclass;
8338     console.log('-- search style rule for '+selector)
8339
8340     for(let i = 0; i < rlen; i++){
8341       cr = ss.cssRules[i];
8342       console.log('CSS rule ['+i+'/'+rlen'] '+cr.selectorText);
8343       if(cr.selectorText === selector){ // css class selector
8344         tabrule = ss.cssRules[i];
8345         console.log('CSS rule found for:['+i+'/'+rlen'] '+selector);
8346         ss.deleteRule(i);
8347         //rlen = ss.cssRules.length;
8348         rulex = i
8349         // should search and replace the property here
8350       }
8351     }
8352     // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
8353     if( tabrule == null ){
8354       console.log('CSS rule NOT found for:[ '+rlen' ] '+selector);
8355       ss.insertRule(cstyle,rlen);
8356       ss.insertRule(cstyle,0); // override by 0?
8357       console.log('CSS rule inserted:[ '+rlen+']\n'+cstyle);
8358     }else{
8359       ss.insertRule(cstyle,rulex);
8360       ss.insertRule(cstyle,0);
8361       console.log('CSS rule replaced:[ '+rulex+']\n'+cstyle);
8362     }
8363   }
8364   function GJE_SetStyle(te){
8365     console.log('Apply the style to:' +te.id+'\n');
8366     console.log('Apply the style to:' +te.parentNode.id+'\n');
8367     console.log('Apply the style to:' +te.parentNode.class+'\n');
8368     cclass = te.parentNode.class;
8369     setCSofClass(cclass,te.value); // should get selector part from
8370           // selector { rules }
8371
8372     if(false){
8373       //console.log('Apply the style:')
8374       //stid = this.parentNode.id+this.id+
8375       //stid = this.id+'.style"
8376       css = te.value
8377       stid = te.parentNode.id+'.style"
8378       e = document.getElementById(stid)
8379       if( e != null ){
8380         //console.log('Apply the style:' +e.id+'\n'+te.value);
8381         console.log('Apply the style:' +e.id+'\n'+css);
8382         //e.innerHTML = css; //te.value;
8383         //ncss = e.sheet;
8384         //ncss.insertRule(te.value,ncss.cssRules.length);
8385       }else{
8386         console.log('No element to Apply the style: '+stid)
8387       }
8388       tblid = te.parentNode.id+"table";
8389       e = document.getElementById(tblid);
8390       if( e != null ){
8391         //e.setAttribute('style',css);
8392         e.setProperty('style',css,'!important');
8393       }
8394     }
8395   }
8396   function makeTable(argv){
8397     //tid = ''
8398     //cwe = GJE_CurElement
8399     cwe = GJCL_Container;
8400     //cwd = GJFactory;
8401     tid = 'table_' + GJE_TableNo
8402
8403     nt = new Text('\n')
8404     cwe.appendChild(nt)
8405
8406     ne = document.createElement('span'); // the container
8407     cwe.appendChild(ne)
8408     ne.id = tid + '-span'
8409     ne.setAttribute('contenteditable',true)
8410
8411     htspan = document.createElement('span'); // html part
8412     //htspan.id = tid + '-html'
8413     //ne.innerHTML = '\n'
8414     nt = new Text('\n')
8415     ne.appendChild(nt)
8416     ne.appendChild(htspan)
8417
8418     htspan.id = tid
8419     htspan.setAttribute('class',tid)
8420
8421     ne.setAttribute('draggable','true')
8422     ne.addEventListener('drag',GJE_DragEvent);
8423     ne.addEventListener('dragend',GJE_DropEvent);
8424
8425     var col = 3
8426     var row = 2
8427     if( argv[0] != null ){
8428       col = argv[0]
8429       argv.shift()
8430     }
8431     if( argv[0] != null ){
8432       row = argv[0]
8433     }

```

```

4433     argv.shift()
4434 }
4435
4436 //ne.setAttribute('class',tid)
4437 ht = "\n"
4438 //ht += '<'+table+' '+id="'+tid+'"'+' class="'+tid+'"''
4439 ht += '<'+table+
4440   + ' onkeydown="GJE_TableKeyCommand(event,this)"'+
4441   //+ ' ondrag="GJE_DragEvent(event,this)"\n'+
4442   //+ ' ondragend="GJE_DropEvent(event,this)"\n'+
4443   //+ ' dragable="true"\n'+
4444   //+ ' contenteditable="true"'.
4445   + '>\n'
4446 ht += '<'+tbody+'>\n';
4447 for( r = 0; r < row; r++ ){
4448   ht += '<'+tr+'>\n'
4449   for( c = 0; c < col; c++ ){
4450     ht += '<'+td+'>\n'
4451     ht += " ABCDEFGHIJKLMNOPQRSTUVWXYZ".charAt(c) + r
4452     ht += '</td>\n'
4453   }
4454   ht += '</'+tbody+'>\n'
4455 }
4456 ht += '<'+table+'>\n';
4457 htspan.innerHTML = ht;
4458 nt = new Text('\n')
4459 ne.appendChild(nt)
4460
4461 st = '#'+tid+' *{\n' // # for instance specific
4462   +' border:1px solid #aaa;\n'
4463   +' background-color:#eef;\n'
4464   +' color:#222;\n'
4465   +' font-size:#14pt !important;\n'
4466   +' font-family:monospace,Courier New !important;\n'
4467   +' }/* hit ESC to apply */\n'
4468
4469 // wish script to be included
4470 //nj = document.createElement('script')
4471 //ne.appendChild(nj)
4472 //ne.innerHTML = 'function SetStyle(e){'
4473
4474 // selector seems lost in dynamic style appending
4475 if(false){
4476   ns = document.createElement('style')
4477   ne.appendChild(ns)
4478   ns.id = tid + '-style'
4479   ns.style_def = ''
4480   ns.innerHTML = '\n'+st
4481   nt = new Text('\n')
4482   ne.appendChild(nt)
4483 }
4484 setCSSOfClass(tid,st); // should be in Javascript script?
4485
4486 nx = document.createElement('textarea')
4487 ne.appendChild(nx)
4488 nx.id = tid + '-style_def'
4489 nx.setAttribute('class','GJ_StyleEditor')
4490 nx.spellcheck = false
4491 nx.cols = 60
4492 nx.rows = 10
4493 nx.innerHTML = '\n'+st
4494 nx.addEventListener('change',GJE_SetTableStyle);
4495 nx.addEventListener('keydown',GJE_StyleKeyCommand);
4496 //nx.addEventListener('click',GJE_SetTableStyle);
4497
4498 nt = new Text('\n')
4499 cwe.appendChild(nt)
4500
4501 GJE_TableNo += 1
4502 return 'created TABLE id="'+tid+''";
4503 }
4504 function GJE_NodeEdit(argv){
4505   cwe = GJE_CurElement
4506   cmd = argv[0]
4507   argv.shift()
4508   args = argv.join(' ')
4509   ret = ""
4510
4511   if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
4512     if( GJE_NodeSaved != null ){
4513       xn = GJE_RootNode
4514       GJE_RootNode = GJE_NodeSaved
4515       GJE_NodeSaved = xn
4516       ret = '-- did undo'
4517     }else{
4518       ret = '-- could not undo'
4519     }
4520   }
4521   return ret
4522 }
4523 GJE_NodeSaved = GJE_RootNode.cloneNode()
4524 if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
4525   if( argv[0] == null ){
4526     ne = GJE_RootNode
4527   }else{
4528     if( argv[0] == '..' ){
4529       ne = cwe.parentNode
4530     }else{
4531       ne = document.getElementById(argv[0])
4532     }
4533     if( ne != null ){
4534       GJE_CurElement = ne
4535       ret = "-- current node: " + ne.id
4536     }else{
4537       ret = "-- not found: " + argv[0]
4538     }
4539   }
4540   if( cmd == '.mkt' || cmd == '.mktable' ){
4541     makeTable(argv)
4542   }
4543   if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
4544     ne = document.createElement(argv[0])
4545     //ne.id = argv[0]
4546     ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
4547     cwe.appendChild(ne)
4548     if( cmd == '.m' || cmd == '.mk' ){
4549       GJE_CurElement = ne
4550     }
4551   }
4552   if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
4553     cwe.id = argv[0]
4554   }
4555   if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
4556   }
4557   if( cmd == '.b' || cmd == '.sh' || cmd == 'sh' ){
4558

```

```

8557     s = argv.join(' ')
8558     cwe.innerHTML = s
8559   }else
8560     if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
8561       cwe.setAttribute(argv[0],argv[1])
8562     }else
8563     if( cmd == '.l' ){
8564     }else
8565     if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
8566       ret = cwe.innerHTML
8567     }else
8568     if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
8569       ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
8570       for(we = cwe.parentNode; we != null; ){
8571         ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
8572         we = we.parentNode
8573       }
8574     }else
8575     {
8576       ret = "Command: mk | rm \n"
8577       ret += "    pw -- print current node\n"
8578       ret += "    mk type -- make node with name and type\n"
8579       ret += "    nm name -- set the id #name of current node\n"
8580       ret += "    rm name -- remove named node\n"
8581       ret += "    cd name -- change current node\n"
8582     }
8583   //alert(ret)
8584   return ret
8585 }
8586 function GJC_Command(text){
8587   lines = text.value.split('\n')
8588   line = lines[lines.length-1]
8589   argv = line.split(' ')
8590   text.value += '\n'
8591   if( argv[0] == '%' ){ argv.shift() }
8592   args0 = argv.join(' ')
8593   cmd = argv[0]
8594   argv.shift()
8595   args = argv.join(' ')
8596
8597   if( cmd == 'nolog' ){
8598     StopConsoleLog = true
8599   }else
8600   if( cmd == 'new' ){
8601     if( argv[0] == 'table' ){
8602       argv.shift()
8603       console.log('argv=' + argv)
8604       text.value += makeTable(argv)
8605     }else
8606     if( argv[0] == 'console' ){
8607       text.value += GJ_NewConsole('GJ_Console')
8608     }else{
8609       text.value += '-- new { console | table }'
8610     }
8611   }else
8612   if( cmd == 'strip' ){
8613     //text.value += GJF_StripClass()
8614   }else
8615   if( cmd == 'css' ){
8616     sel = '#table_1'
8617     if(argv[0]==0')
8618       rule1 = sel+'{color:#000 !important; background-color:#fff !important;}' ;
8619     else
8620       rule1 = sel+'{color:#f00 !important; background-color:#eef !important;}' ;
8621     document.styleSheets[3].deleteRule(0);
8622     document.styleSheets[3].insertRule(rule1,0);
8623     text.value += 'CSS rule added: '+rule1
8624   }else
8625   if( cmd == 'print' ){
8626     e = null;
8627     if( e == null ){
8628       e = document.getElementById('GJFactory_0')
8629     }else
8630     if( e == null ){
8631       e = document.getElementById('GJFactory_1')
8632     }
8633     if( argv[0] != null ){
8634       id = argv[0]
8635       if( id == 'f' ){
8636         //e = document.getElementById('GJE_RootNode');
8637       }else{
8638         e = document.getElementById(id)
8639       }
8640       if( e != null ){
8641         text.value += e.outerHTML
8642       }else{
8643         text.value += "Not found: " + id
8644       }
8645     }else{
8646       text.value += GJE_RootNode.outerHTML
8647     //text.value += e.innerHTML
8648   }
8649 }else
8650   if( cmd == 'destroy' ){
8651     text.value += GJFactory_Destroy()
8652   }else
8653   if( cmd == 'save' ){
8654     e = document.getElementById('GJFactory')
8655     Permanent.setItem('GJFactory-1',e.innerHTML)
8656     text.value += "-- Saved GJFactory"
8657   }else
8658   if( cmd == 'load' ){
8659     gif = Permanent.getItem('GJFactory-1')
8660     e = document.getElementById('GJFactory')
8661     e.innerHTML = gif
8662     // must restore EventListener
8663     text.value += "-- EventListener was not restored"
8664   }else
8665   if( cmd.charAt(0) == '.' ){
8666     argv0 = args0.split('.')
8667     text.value += GJE_NodeEdit(argv0)
8668   }else
8669   if( cmd == 'cont' ){
8670     bannerIsStopping = false
8671     GshMenuStop.innerHTML = "Stop"
8672   }else
8673   if( cmd == 'date' ){
8674     text.value += DateLong()
8675   }else
8676   if( cmd == 'echo' ){
8677     text.value += args
8678   }else
8679   if( cmd == 'fork' ){
8680     html_fork()

```

```

8681     }else
8682     if( cmd == 'last' ){
8683         text.value += MyHistory
8684         //h = document.createElement("span")
8685         //h.innerHTML = MyHistory
8686         //text.value += h.innerHTML
8687         //tx = MyHistory.replace("\n","");
8688         //text.value += tx.replace("<"+>br>","\n") + "xxxx<"+>br>yyyy"
8689     }else
8690     if( cmd == 'ne' ){
8691         text.value += GJE_NodeEdit(argv)
8692     }else
8693     if( cmd == 'reload' ){
8694         location.reload()
8695     }else
8696     if( cmd == 'mem' ){
8697         text.value += GJC_Memory('GJC_Storage',args,text)
8698     }else
8699     if( cmd == 'stop' ){
8700         bannerIsStopping = true
8701         GshMenuStop.innerHTML = "Start"
8702     }else
8703     if( cmd == 'who' ){
8704         text.value += "SessionId="+GJC_SessionId+" "+document.URL
8705     }else
8706     if( cmd == 'wall' ){
8707         text.value += GJC_Memory('GJC_Wall','write',text)
8708     }else
8709     {
8710         text.value += "Commands: help | echo | date | last \n"
8711         + '          new | save | load | mem \n'
8712         + '          who | wall | fork | nife'
8713     }
8714 }
8715
8716 function GJC_Input(){
8717     if( this.value.endsWith("\n") ){ // remove NL added by textarea
8718         this.value = this.value.slice(0,this.value.length-1)
8719     }
8720 }
8721
8722 var GJC_Id = null
8723 function GJC_Resize(){
8724     GJC_Id.style.zIndex = 20000
8725     //GJC_Id.style.width = window.innerWidth - 16
8726     GJC_Id.style.width = '100%';
8727     GJC_Id.style.height = 300;
8728     GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
8729     GJC_Id.style.color = "rgba(255,255,255,1.0)"
8730 }
8731 function GJC_FocusIn(){
8732     this.spellcheck = false
8733     SuppressGJShell = true
8734     this.onkeydown = GJC_KeyDown
8735     GJC_Resize()
8736 }
8737 function GJC_FocusOut(){
8738     SuppressGJShell = false
8739     this.removeEventListener('keydown',GJC_KeyDown);
8740 }
8741 window.addEventListener('resize',GJC_Resize);
8742
8743 function GJC_OnStorage(e){
8744     //alert('Got Message')
8745     //GJC.value += "\n((ReceivedMessage))\n"
8746 }
8747 window.addEventListener('storage',GJC_OnStorage);
8748 //window.addEventListener('storage',()=>(alert('GotMessage')))
8749
8750 function GJC_Setup(gjcid){
8751     //gjcid.style.width = gsh.getBoundingClientRect().width
8752     gjcid.style.width = '100%';
8753     gjcid.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
8754     //gjcid.value += "Date: " + DateLong() + "\n"
8755     gjcid.value += PS1
8756     gjcid.onfocus = GJC_FocusIn
8757     gjcid.addEventListener('input',GJC_Input);
8758     gjcid.addEventListener('focusout',GJC_FocusOut);
8759     GJC_Id = gjcid
8760 }
8761 function GJC_Clear(id){
8762 }
8763 function GJConsole_initConsole(){
8764     if( document.getElementById("GJC_0") != null ){
8765         GJC_Setup(GJC_0)
8766     }else{
8767         GJC_Container.innerHTML = '<'
8768         +`<'+`/textarea>`;
8769         GJC_Setup(GJC_1)
8770         factory = document.createElement('span');
8771         gsh.appendChild(factory)
8772         GJE_RootNode = factory;
8773         GJE_CurElement = GJE_RootNode;
8774     }
8775     var initGJCF = false;
8776     function GJConsole_initFactory(){
8777         if( initGJCF ){ return; } initGJCF = true;
8778         GShell_initKeyCommands();
8779         GJConsole_initConsole();
8780     }
8781 }
8782 //GJConsole_initFactory();
8783 // TODO: focus handling
8784 </script>
8785 <style>
8786 .GJ_StyleEditor {
8787     font-size:9pt !important;
8788     font-family:Courier New, monospace !important;
8789 }
8790 </style>
8791 <details>
8792 </span>
8793 <!-- ----- GJConsole END } ----- -->
8794 */
8795
8796 /*
8797 <span id="BlinderText">
8798 <style id="BlinderTextStyle">
8800 #GJLinkView {
8801     position: absolute; z-index:5000;
8802     position: relative;
8803     display: block;
8804     left: 8px;

```

```

8805     color:#ffff;
8806     width:800px; height:300px; resize:both;
8807     margin:0px; padding:4px;
8808     background-color:rgba(200,200,200,0.5) !important;
8809   }
8810 .MsgText {
8811   width:578px !important;
8812   resize:both !important;
8813   color:#000 !important;
8814 }
8815 .GjNote {
8816   font-family:Georgia !important;
8817   font-size:13pt !important;
8818   color:#22a !important;
8819 }
8820 .textField {
8821   display:inline;
8822   border:0.5px solid #444;
8823   border-radius:3px;
8824   color:#000; background-color:#fff;
8825   width:106pt; height:18pt;
8826   margin:2px;
8827   padding:2px;
8828   resize:none;
8829   vertical-align:middle;
8830   font-size:10pt; font-family:Courier New;
8831 }
8832 .textLabel {
8833   border:0px solid #000 !important;
8834   background-color:rgba(0,0,0,0);
8835 }
8836 .textURL {
8837   width:300pt !important;
8838   border:0px solid #000 !important;
8839   background-color:rgba(0,0,0,0);
8840 }
8841 .VisibleText {
8842 }
8843 .BlinderText {
8844   color:#000; background-color:#eee;
8845 }
8846 .joinButton {
8847   font-family:Georgia !important;
8848   font-size:11pt;
8849   line-height:1.1;
8850   height:18pt;
8851   width:50pt;
8852   padding:3px !important;
8853   text-align:center !important;
8854   border-color:#aaa !important;
8855   border-radius:5px;
8856   color:#fff; background-color:#4a4 !important;
8857   vertical-align:middle !important;
8858 }
8859 .SendButton {
8860   vertical-align:top;
8861 }
8862 .ws0_log {
8863   font-size:10pt;
8864   color:#000 !important;
8865   line-height:1.0;
8866   background-color:rgba(255,255,255,0.7) !important;
8867   font-family:Courier New,monospace !important;
8868   width:99.3%;
8869   white-space:pre;
8870 }
8871 </style>
8872
8873 <!-- Form autofill test
8874 Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafrm/formLogin" size="80">
8875 <form method="POST" id="xxform" action="https://192.168.10.1/boafrm/formLogin">
8876 dest?    <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
8877 -->
8878 <details><summary>Form Auto. Filling</summary>
8879 <style>
8880 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
8881   display:inline !important; font-size:10pt !important; padding:1px !important;
8882 }
8883 </style>
8884 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
8885 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
8886 Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
8887 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
8888 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
8889 SessionId: <input id="xxsid" class="xxinput" name="SESSION_ID" type="text" size="80">
8890 <input id="xxsub" class="xxinput" type="submit" value="Submit"></form>
8891 </span>
8892 <script>
8893   function XXSetFormAction(){
8894     xxform.setAttribute('action',xxserv.value);
8895   }
8896   xxform.setAttribute('action',xxserv.value);
8897   xxserv.addEventListener('change',XXSetFormAction);
8898 //xxserv.value = location.href;
8899 </script>
8900 </details>
8901 */
8902 /*
8903 <!--
8904 <details id="BlinderTextClass" class="gsh-src"><summary>BlinderText</summary>
8905 <span id="BlinderTextScript">
8906 // https://w3c.github.io/uievents/#event-type-keydown
8907 //
8908 // 2020-09-21 class BlinderText - textarea element not to be readable
8909 //
8910 // BlinderText attributes
8911 // bl_plainText - null
8912 // bl_hideChecksum - [false]
8913 // bl_showLength - [false]
8914 // bl_visible - [false]
8915 // data-bl_config - []
8916 //   - min. length
8917 //   - max. length
8918 //   - acceptable charset in generete text
8919 //
8920 function BlinderChecksum(text){
8921   plain = text.bl_plainText;
8922   return strCRC32(plain,plain.length).toFixed(0);
8923 }
8924 function BlinderKeydown(ev){
8925   pass = ev.target
8926   if( ev.code == 'Enter' ){
8927     ev.preventDefault();
8928   }

```

```

8929     ev.stopPropagation()
8930   }
8931   function BlinderKeyup(ev){
8932     blind = ev.target
8933     if( ev.code == 'Backspace'){
8934       blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
8935     }else
8936     if( (and(ev.code == 'KeyV', ev.ctrlKey) ){
8937       blind.bl_visible = !blind.bl_visible;
8938     }else
8939     if( (and(ev.code == 'KeyL', ev.ctrlKey) ){
8940       blind.bl_showLength = !blind.bl_showLength;
8941     }else
8942     if( (and(ev.code == 'KeyU', ev.ctrlKey) ){
8943       blind.bl_plainText = "";
8944     }else
8945     if( (and(ev.code == 'KeyR', ev.ctrlKey) ){
8946       checksum = BlinderChecksum(blind);
8947       blind.bl_plainText = checksum; // .toString(32);
8948     }else
8949     if( ev.code == 'Enter' ){
8950       ev.stopPropagation();
8951       ev.preventDefault();
8952       return;
8953     }else
8954     if( ev.key.length != 1 ){
8955       console.log('KeyUp: '+ev.code+'/'+ev.key);
8956       return;
8957     }else{
8958       blind.bl_plainText += ev.key;
8959     }
8960   leng = blind.bl_plainText.length;
8961   //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
8962   checksum = BlinderChecksum(blind) % 10; // show last one digit only
8963
8964   visual = '';
8965   if( !blind.bl_hideCheckSum || blind.bl_showLength ){
8966     visual += '[';
8967   }
8968   if( !blind.bl_hideCheckSum ){
8969     visual += '#'+checksum.toString(10);
8970   }
8971   if( blind.bl_showLength ){
8972     visual += '/' + leng;
8973   }
8974   if( !blind.bl_hideCheckSum || blind.bl_showLength ){
8975     visual += ']';
8976   }
8977   if( blind.bl_visible ){
8978     visual += blind.bl_plainText;
8979   }else{
8980     visual += '.'.repeat(leng);
8981   }
8982 }
8983 blind.value = visual;
8984 }
8985 function BlinderKeyup(ev){
8986   BlinderKeyup(ev);
8987   ev.stopPropagation();
8988 }
8989 // https://w3c.github.io/uievents/#keyboardevent
8990 // https://w3c.github.io/uievents/#uievent
8991 // https://dom.spec.whatwg.org/#event
8992 function BlinderTextEvent(){
8993   ev = event;
8994   blind = ev.target;
8995   console.log('Event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
8996   if( ev.type == 'keyup' ){
8997     BlinderKeyup(ev);
8998   }else
8999   if( ev.type == 'keydown' ){
9000     BlinderKeydown(ev);
9001   }else{
9002     console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
9003   }
9004 }
9005 //< textarea hidden id="BlinderTextClassDef" class="textField"
9006 // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
9007 // spellcheck="false"></textarea>
9008 //< textarea hidden id="gj_pass1"
9009 // class="textField BlinderText"
9010 // placeholder="PassWord1"
9011 // onkeydown="BlinderTextEvent()"
9012 // onkeyup="BlinderTextEvent()"
9013 // spellcheck="false"></textarea>
9014 function SetupBlinderText(parent,txa,phold){
9015   if( txa == null ){
9016     txa = document.createElement('textarea');
9017     //txa.id = id;
9018   }
9019   txa.setAttribute('class','textField BlinderText');
9020   txa.setAttribute('placeholder',phold);
9021   txa.setAttribute('onkeydown','BlinderTextEvent()');
9022   txa.setAttribute('onkeyup','BlinderTextEvent()');
9023   txa.setAttribute('spellcheck','false');
9024   //txa.setAttribute('bl_plainText','false');
9025   txa.bl_plainText = '';
9026   //parent.appendChild(txa);
9027 }
9028 function DestroyBlinderText(txa){
9029   txa.removeAttribute('class');
9030   txa.removeAttribute('placeholder');
9031   txa.removeAttribute('onkeydown');
9032   txa.removeAttribute('onkeyup');
9033   txa.removeAttribute('spellcheck');
9034   txa.bl_plainText = '';
9035 }
9036 //
9037 // visible textarea like Username
9038 //
9039 function VisibleTextEvent(){
9040   if( event.code == 'Enter' ){
9041     if( event.target.NoEnter ){
9042       event.preventDefault();
9043     }
9044   }
9045   event.stopPropagation();
9046 }
9047 function SetupVisibleText(parent,txa,phold){
9048   if( false ){
9049     txa.setAttribute('class','textField VisibleText');
9050   }else{
9051     newclass = txa.getAttribute('class');
9052     if( and(newclass != null, newclass != '' ) ){

```

```
9053     newclass += ' ';
9054   }
9055   newclass += 'VisibleText';
9056   txa.setAttribute('class',newclass);
9057 }
9058 //console.log('SetupVisibleText class='+txa.class);
9059 txa.setAttribute('placeholder',phold);
9060 txa.setAttribute('onkeydown','VisibleTextEvent()');
9061 txa.setAttribute('onkeyup', 'VisibleTextEvent()');
9062 txa.setAttribute('spellcheck','false');
9063 cols = txa.getAttribute('cols');
9064 if( cols != null ){
9065   txa.style.width = '580px';
9066   //console.log('VisualText#'+txa.id+' cols='+cols)
9067 }else{
9068   //console.log('VisualText#'+txa.id+' NO cols')
9069 }
9070 rows = txa.getAttribute('rows');
9071 if( rows != null ){
9072   txa.style.height = '30px';
9073   txa.style.resize = 'both';
9074   txa.NoEnter = false;
9075 }else{
9076   txa.NoEnter = true;
9077 }
9078 }
9079 function DestroyVisibleText(txa){
9080   txa.removeAttribute('class');
9081   txa.removeAttribute('placeholder');
9082   txa.removeAttribute('onkeydown');
9083   txa.removeAttribute('onkeyup');
9084   txa.removeAttribute('spellcheck');
9085   cols = txa.removeAttribute('cols');
9086 }
9087 </span>
9088 <script>
9089 js = document.getElementById('BlinderTextScript');
9090 eval(js.innerHTML);
9091 //js.outerHTML = ""
9092 </script>
9093
9094 </details>
9095 </span>
9096 */
9097
9098 /*
9099 <script id="GJLinkScript">
9100 function gjkey_hash(text){
9101   return strCRC32(text,text.length) % 0x10000;
9102 }
9103 function gj_addlog(e,msg){
9104   now = (new Date().getTime() / 1000).toFixed(3);
9105   tstamp = '['+now+']';
9106   e.value += tstamp + msg;
9107   e.scrollTop = e.scrollHeight;
9108 }
9109 function gj_addlog_c1(msg){
9110   ws0_log.value += '(console.log) ' + msg + '\n';
9111 }
9112 var GJ_Channel = null;
9113 var GJ_Log = null;
9114 var gjx; // the global variable
9115 function GJ_Join(){
9116   target = gj_join;
9117   if( target.value == 'Leave' ){
9118     GJ_Channel.close();
9119     GJ_Channel = null;
9120     target.value = 'Join';
9121     return;
9122   }
9123
9124 var ws0;
9125 var ws0_log;
9126
9127 sav_console_log = console.error
9128 console.error = gj_addlog_c1
9129 ws0 = new WebSocket(gj_serv.innerHTML);
9130 console.error = sav_console_log
9131
9132 GJ_Channel = ws0;
9133 ws0_log = document.getElementById('ws0_log');
9134 GJ_Log = ws0_log;
9135
9136 now = (new Date().getTime() / 1000).toFixed(3);
9137 const wsstats = ["CONNECTING", "OPEN", "CLOSING", "CLOSED"];
9138 cst = wsstats[ws0.readyState];
9139 gj_addlog(ws0_log, stat +'ws0.readyState'+': GJ Linked\n');
9140
9141 ws0.addEventListener('error', function(event){
9142   gj_addlog(ws0_log, stat + 'error : transport error?\n');
9143 });
9144 ws0.addEventListener('open', function(event){
9145   GJLinkView.style.zIndex = 10000;
9146   //console.log('#'+GJLinkView.id+'.zIndex=' +GJLinkView.style.zIndex);
9147   date1 = new Date().getTime();
9148   date2 = (date1 / 1000).toFixed(3);
9149   seed = date1.toString(16);
9150
9151   // user name and key
9152   user = document.getElementById('gj_user').value;
9153   if( user.length == 0 ){
9154     gj_user.value = 'nemo';
9155     user = 'nemo';
9156   }
9157   key1 = document.getElementById('gj_ukey').bl_plainText;
9158   ukey = gjkey_hash(seed+user+key1).toString(16);
9159
9160   // session name and key
9161   chan = document.getElementById('gj_chan').value;
9162   if( chan.length == 0 ){
9163     gj_chan.value = 'main';
9164     chan = 'main';
9165   }
9166   key2 = document.getElementById('gj_ckey').bl_plainText;
9167   ckey = gjkey_hash(seed+chan+key2).toString(16);
9168
9169   msg = date2 + ' JOIN ' + user + '!' + chan + ' ' + ukey + ':' + ckey;
9170   gj_addlog(ws0_log,'send '+msg+'\n');
9171   ws0.send(msg);
9172
9173   target.value = 'Leave';
9174   //console.log('[+date2+] #' +target.id+ ' '+target.value+'\n');
9175   //gj_addlog(ws0_log,'label '+target.value+'\n');
9176 });

});
```

```

9177 ws0.addEventListener('message', function(event){
9178   now = (new Date().getTime() / 1000).toFixed(3);
9179   msg = event.data;
9180   gj_addlog(ws0_log,'recv '+msg+'\n');
9181
9182   argv = msg.split(' ')
9183   tstamp = argv[0];
9184   argv.shift();
9185   if( argv[0] == 'reload' ){
9186     location.reload()
9187   }
9188   argv.shift(); // command
9189   argv.shift(); // from|to
9190   if( argv[0] == 'auth' ){
9191     // doing authorization required
9192   }
9193   if( argv[0] == 'echo' ){
9194     now = (new Date().getTime() / 1000).toFixed(3);
9195     msg = now+' '+RESP 'argv.join(' ');
9196     gj_addlog(ws0_log,'send '+msg+'\n');
9197     ws0.send(msg);
9198   }
9199   if( argv[0] == 'eval' ){
9200     argv.shift();
9201     js = argv.join(' ');
9202     ret = eval(js); // ----- eval()
9203     gj_addlog(ws0_log,'eval '+js+' = '+ret+'\n');
9204     now = (new Date().getTime() / 1000).toFixed(3);
9205     msg = now+' '+RESP ' + ret;
9206     ws0.send(msg);
9207     gj_addlog(ws0_log,'send '+msg+'\n')
9208   }
9209 });
9210 ws0.addEventListener('close', function(event){
9211   if( GJ_Channel == null ){
9212     gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
9213     return;
9214   }
9215   GJ_Channel.close();
9216   GJ_Channel = null;
9217   target.value = 'Join';
9218   gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
9219 });
9220 }
9221 function GJ_SendMessageUserPass(user,chan,msgbody){
9222   now = (new Date().getTime() / 1000).toFixed(3);
9223   msg = now+' ISAY '+user+'|'+chan+'|'+msgbody;
9224   gj_addlog(GJ_Log,'send '+msg+'\n');
9225   GJ_Channel.send(msg);
9226 }
9227 function GJ_SendMessage(msgbody){
9228   if( GJ_Channel == null ){
9229     gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
9230     return;
9231   }
9232   //target = event.target;
9233   user = document.getElementById('gj_user').value;
9234   chan = document.getElementById('gj_chan').value;
9235   GJ_SendMessageUserPass(user,chan,msgbody);
9236 }
9237 function GJ_Send(){
9238   msgbody = gj_sendText.value;
9239   GJ_SendMessage(msgbody);
9240 }
9241 </script>
9242
9243 <!-- ----- GJLINK ----- -->
9244 <!--
9245   - User can subscribe to a channel
9246   - A channel will be broadcasted
9247   - A channel can be a pattern (regular expression)
9248   - User is like From:(me) and channel is like To: or Recipient:
9249   - like VIBUS
9250     - watch message with SENDME, WATCH, CATCH, HEAR, or so
9251     - routing with path expression or name pattern (with routing with DNS like system)
9252 -->
9253 */
9254
9255 //<span id="GJLinkGolang">
9256 //<details id="GshWebSocket" class="gsh-src"><summary>Golang / JavaScript Link</summary>
9257 // 2020-0920 created
9258 // <a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
9259 // <a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
9260 // INSTALL: go get golang.org/x/net/websocket
9261 // INSTALL: sudo {apt,yum} install git (if git is not installed yet)
9262 // import "golang.org/x/net/websocket"
9263 const gshws_origin = "http://localhost:9999"
9264 const gshws_server = "localhost:9999"
9265 const gshws_port = 9999
9266 const gshws_path = "gjlink1"
9267 const gshws_url = "ws://" + gshws_server + "/" + gshws_path
9268 const GSHWS_MSGSIZE = (8*1024)
9269 func fmtstring(fmts string, params ...interface{})(string){
9270   return fmt.Sprintf(fmts,params...)
9271 }
9272 func GSHWS_MARK(what string)(string){
9273   now := time.Now()
9274   us := fmtstring("%06d",now.Nanosecond() / 1000)
9275   mark := ""
9276   if( !AtConsoleLineTop ){
9277     mark += "\n"
9278     AtConsoleLineTop = true
9279   }
9280   mark += "["+now.Format(time.Stamp)+"."+us+"] -GJ- " + what + ": "
9281   return mark
9282 }
9283 func gchk(what string,err error){
9284   if( err != nil ){
9285     panic(GSHWS_MARK(what)+err.Error())
9286   }
9287 }
9288 func glog(what string, fmts string, params ...interface{}{
9289   fmt.Println(GSHWS_MARK(what))
9290   fmt.Printf(fmts+"\n",params...)
9291 }
9292
9293 var WSV = []*websocket.Conn{}
9294 func jsend(argv []string){
9295   if len(argv) <= 1 {
9296     fmt.Printf("--Ij %v [-m] command arguments\n",argv[0])
9297     return
9298   }
9299   argv = argv[1:]
9300   if( len(WSV) == 0 ){

```

```

9301     fmt.Printf("--Ej-- No link now\n")
9302     return
9303 }
9304 if( 1 < len(WSV) ){
9305     fmt.Printf("--Ij-- multiple links (%v)\n",len(WSV))
9306 }
9307
9308 multicast := false // should be filtered with regexp
9309 if( 0 < len(argv) && argv[0] == "-m" ){
9310     multicast = true
9311     argv = argv[1:]
9312 }
9313 args := strings.Join(argv," ")
9314
9315 now := time.Now()
9316 msec := now.UnixNano() / 1000000;
9317 tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9318 msg := fmtstring("%v SEND gshell|* %v",tstamp,args)
9319
9320 if( multicast ){
9321     for i,ws := range WSV {
9322         wn,werr := ws.Write([]byte(msg))
9323         if( werr != nil ){
9324             fmt.Printf("(%v) wn=%v, werr=%v\n",i,wn,werr)
9325         }
9326         glog("SQ",fmtstring("(%v) %v",wn,msg))
9327     }
9328 }else{
9329     i := 0
9330     ws := WSV[i]
9331     wn,werr := ws.Write([]byte(msg))
9332     if( werr != nil ){
9333         fmt.Printf("(%v) wn=%v, werr=%v\n",i,wn,werr)
9334     }
9335     glog("SQ",fmtstring("(%v) %v",wn,msg))
9336 }
9337 }
9338 func servl(ws *websocket.Conn) {
9339     WSV = append(WSV,ws)
9340     //fmt.Println("n")
9341     glog("CO","accepted connections(%v)",len(WSV))
9342     //remoteAddr := ws.RemoteAddr
9343     //fmt.Println("-- accepted %v\n",remoteAddr)
9344     //fmt.Println("-- accepted %v\n",ws.Config())
9345     //fmt.Println("-- accepted %v\n",ws.Config().Header)
9346     //fmt.Println("-- accepted %v // %v\n",ws,servl)
9347
9348     var reqb = make([]byte,GSHWS_MSGSIZE)
9349     for {
9350         rn, rerr := ws.Read(reqb)
9351         if( rerr != nil || rn < 0 ){
9352             glog("SQ",fmtstring("(%v,%v)",rn,rerr))
9353             break
9354         }
9355         req := string(reqb[0:rn])
9356         glog("SQ",fmtstring("(%v) %v",rn,req))
9357
9358         argv := strings.Split(req, " ");
9359         argv = argv[1:]
9360         if( 0 < len(argv) ){
9361             if( argv[0] == "RESP" ){
9362                 // should forward to the destination
9363                 continue;
9364             }
9365         }
9366         now := time.Now()
9367         msec := now.UnixNano() / 1000000;
9368         tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9369         res := fmtstring("%v "+CAST+" %v",tstamp,req)
9370         wn, werr := ws.Write([]byte(res))
9371         gchck("SR",werr)
9372         glog("SR",fmtstring("(%v) %v",wn,string(res)))
9373     }
9374     glog("SF","WS response finish")
9375
9376     wsv := []*websocket.Conn{}
9377     wsx := 0
9378     for i,v := range WSV {
9379         if( v != ws ){
9380             wsx = i
9381             wsv = append(wsv,v)
9382         }
9383     }
9384     WSV = wsv
9385     //glog("CO","closed %v",ws)
9386     glog("CO","closed connection [%v/%v]",wsx+1,len(WSV)+1)
9387     ws.Close()
9388 }
9389 // url ::= [scheme://]host[:port][/:path]
9390 func decomp_URL(url string){
9391 }
9392 func full_wsURL(){
9393 }
9394 func gj_server(argv []string) {
9395     gjiserv := gshws_url
9396     gjport := gshws_server
9397     gjpath := gshws_path
9398     gjischeme := "ws"
9399
9400     //cmd := argv[0]
9401     argv = argv[1:]
9402     if( 1 <= len(argv) ){
9403         serv := argv[0]
9404         if( 0 < strings.Index(serv,"://") ){
9405             schemev := strings.Split(serv,"://")
9406             gjischeme = schemev[0]
9407             serv = schemev[1]
9408         }
9409         if( 0 < strings.Index(serv,"/") ){
9410             pathv := strings.Split(serv,"/")
9411             serv = pathv[0]
9412             gjpath = pathv[1]
9413         }
9414         servv := strings.Split(serv,":")
9415         host := "localhost"
9416         port := 9999
9417         if( servv[0] != "" ){
9418             host = servv[0]
9419         }
9420         if( len(servv) == 2 ){
9421             fmt.Sscanf(servv[1],"%d",&port)
9422         }
9423         //glog("IC","hostport=%v (%v : %v)",servv,host,port)
9424         gjport = fmt.Sprintf("%v:%v",host,port)

```

```

9425     gjserv = gjscheme + "://" + gjport + "/" + gjpath
9426   }
9427   glog("LS",fmtstring("listening at %v",gjserv))
9428   http.Handle("/"+gjpath,websocket.Handler(serv1))
9429   err := error(nil)
9430   if( gjscheme == "ws" ){
9431     // https://golang.org/pkg/net/http/#ListenAndServeTLS
9432     //err = http.ListenAndServeTLS(gjport,nil)
9433   }else{
9434     err = http.ListenAndServe(gjport,nil)
9435   }
9436   gchk("LE",err)
9437 }
9438
9439 func gj_client(argv []string {
9440   glog("CS",fmtstring("connecting to %v",gshws_url))
9441   ws, err := websocket.Dial(gshws_url,"",gshws_origin)
9442   gchk("C",err)
9443
9444   var resb = make([]byte, GSHWS_MSGSIZE)
9445   for qi := 0; qi < 3; qi++ {
9446     req := fmtstring("Hello, GShell! (%v)",qi)
9447     wn, werr := ws.Write([]byte(req))
9448     glog("QM",fmtstring("(%v) %v",wn,req))
9449     gchk("QE",werr)
9450     rn, rerr := ws.Read(resb)
9451     gchk("RE",rerr)
9452     glog("RM",fmtstring("(%v) %v",rn,string(resb)))
9453   }
9454   glog("CF","WS request finish")
9455 }
9456 //</details></span>
9457 */
9458 <details><summary>GJ Link</summary>
9459 <span id="GJLinkView" class="GJLinkView">
9460   <p>
9461     <note class="GjNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
9462   </p>
9463
9464   <span id="GJLink_1">
9465     <div id="GJLink_ServerSet"></div>
9466
9467     <div>
9468       <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
9469       <span id="GJLink_Account"></span>
9470     </div>
9471
9472     <div>
9473       <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
9474       <span id="GJLink_SendArea"></span>
9475     </div>
9476
9477     <div id="ws0_log_container"></div>
9478   </span>
9479 </span>
9480
9481 <script>
9482   function setupGJLinkArea(){
9483     GJLink_ServerSet.innerHTML = '<'+span id="gj_serv_label"'+
9484       + ' class="textField textlabel">Server: <'+span>'+
9485       + '<'+span id="gj_serv" class="textField textURL" contenteditable><'+span'>';
9486
9487     GJLink_Account.innerHTML = '<'+textarea id="gj_user" class="textField"><'/textarea>' +
9488       + '<'+textarea id="gj_ukey" class="textField"><'/textarea>' +
9489       + '<'+textarea id="gj_chan" class="textField"><'/textarea>' +
9490       + '<'+textarea id="gj_ckey" class="textField"><'/textarea>';
9491
9492     GJLink_SendArea.innerHTML =
9493       '<'+textarea id="gj_sendText" class="textField MssgText" cols=60 rows=2><'/textarea>';
9494
9495     ws0_log_container.innerHTML = '<'+textarea id="ws0_log" class="ws0_log"'+
9496       + ' cols=100 rows=10 spellcheck="false"><'/textarea>';
9497   }
9498
9499   function clearGJLinkArea(){
9500     GJLink_ServerSet.innerHTML = "";
9501     GJLink_Account.innerHTML = "";
9502     GJLink_SendArea.innerHTML = "";
9503     ws0_log_container.innerHTML = "";
9504   }
9505 </script>
9506
9507 <script>
9508   function SetupGJLink(){
9509     setupGJLinkArea();
9510     SetupVisibleText(GJLink_1,gj_serv,'GJLinkSv');
9511     SetupVisibleText(GJLink_1,gj_user,'UserName');
9512     SetupBlinderText(GJLink_1,gj_ukey,'UserKey');
9513     SetupVisibleText(GJLink_1,gj_chan,'ChannelName');
9514     SetupBlinderText(GJLink_1,gj_ckey,'ChannelKey');
9515     SetupVisibleText(GJLink_1,gj_sendText,'Message');
9516     gj_serv.innerHTML = 'ws://localhost:9999/gjlink1'
9517   }
9518   function GJLink_init(){
9519     SetupGJLink();
9520   }
9521   function iselem(eid){
9522     return document.getElementById(eid);
9523   }
9524   function DestroyGJLink1(){
9525     clearGJLinkArea();
9526     if( !islem('gj_user') ){
9527       return;
9528     }
9529     if( gj_serv_label.parentNode != gj_user ){
9530       return;
9531     }
9532     if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
9533     if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
9534     if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
9535     if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
9536     if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
9537     if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
9538     if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
9539     if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
9540   }
9541   DestroyGJLink = DestroyGJLink1;
9542 </script>
9543 </details>
9544 */
9545
9546 /**
9547 <script id="HtmlCodeview-script">
9548   function showNodeAsHtmlSource(otxa,code){

```

```

9549   txa = document.createElement('textare');
9550   txa.id = otxa.id;
9551   txa.setAttribute('class','HtmlCodeviewText');
9552   otxa.parentNode.replaceChild(txa,otxa);
9553   txa.setAttribute('spellcheck','false');
9554   //txa.value = code.innerHTML;
9555   //txa.innerHTML = code.innerHTML;
9556   txa.innerHTML = code.outerHTML;
9557   txa.style.display = "block";
9558   txa.style.width = "100%";
9559   txa.style.height = "300px";
9560 }
9561 function showHtmlCode(otxa,code){
9562   if( event.target.value == 'ShowCode' ){
9563     showNodeAsHtmlSource(otxa,code);
9564     event.target.value = 'HideCode';
9565   }else{
9566     otxa.style.display = "none";
9567     event.target.value = 'ShowCode';
9568   }
9569 }
9570 </script>
9571 <style id="HtmlCodeview-style">
9572   .HtmlCodeviewText {
9573     font-size:10pt;
9574     font-family:Courier New;
9575     white-space:pre;
9576   }
9577   .HtmlCodeViewButton {
9578     padding:2pt !important;
9579     line-height:1.1 !important;
9580     border:2px inset #bbb !important;
9581     font-size:10pt !important;
9582     font-weight:normal !important;
9583     font-family:Georgia !important;
9584     border-radius:3px !important;
9585     color:#ddd; background-color:#228 !important;
9586   }
9587 </style>
9588 */
9589 */
9590 /*
9591 <details><summary>Live HTML Snapshot</summary>
9592 <span id="LiveHTML">
9593   <!-- ----- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
9594   <input class="HtmlCodeviewButton" type="button" value="ShowCode" onclick="showLiveHTMLcode()">
9595   <span id="LiveHTML_Codeview"></span>
9596   <script id="LiveHTML_Script">
9597     function showLiveHTMLcode(){
9598       showHtmlCode(LiveHTML_Codeview,LiveHTML);
9599     }
9600     var _editable = false;
9601     var savSuppressGJShell = false;
9602     function ToggleEditMode(){
9603       _editable = !_editable;
9604       if( _editable ){
9605         SavSuppressGJShell = SuppressGJShell;
9606         SuppressGJShell = true;
9607         gsh.setAttribute('contenteditable','true');
9608         GshMenuEdit.innerHTML = 'Lock';
9609         GshMenuEdit.style.color = 'rgba(255,0,0,1)';
9610         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
9611     }else{
9612       SuppressGJShell = savSuppressGJShell;
9613       gsh.setAttribute('contenteditable','false');
9614       GshMenuEdit.innerHTML = 'Edit';
9615       GshMenuEdit.style.color = 'rgba(16,160,16,1)';
9616       GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
9617     }
9618   }
9619   function html_edit(){
9620     ToggleEditMode();
9621   }
9622   // Live HTML (DOM) Snapshot onto browser's localStorage
9623   // 2020-0923 SatoxITS
9624   var htRoot = gsh // -- Element-ID, should be selectable
9625   const snappedHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStogate
9626   const htVersionTag = 'VersionTag'; // VersionTag Ellement-ID in the HTML (in DOM)
9627   // -- should be a [map] of URL
9628   const htVersionTag = 'VersionTag'; // VersionTag Ellement-ID in the HTML (in DOM)
9629   function showVersion(note,w,v,u,t){
9630     w.alert(note+' : ' + v + '\n'
9631     + '-- URL: ' + u + '\n'
9632     + '-- Time: ' + t + ' (' + DateLong0(t*1000) + ')';
9633   };
9634 }
9635 }
9636 function html_save(){
9637   u = document.URL;
9638   t = new Date().getTime() / 1000;
9639   v = '<'+span id="'+htVersionTag+'" data-url="'+u+'" data-time="'+t+'>';
9640   v += '<'+span+'\n';
9641   h += v + htRoot.outerHTML;
9642   localStorage.setItem(snappedHTML,h);
9643   showVersion("Saved",window,v,u,t);
9644 }
9645 function html_load(){
9646   h = localStorage.getItem(snappedHTML);
9647   if( h == null ){
9648     alert('No snapshot taken yet');
9649     return;
9650   }
9651   w = window.open('','','');
9652   d = w.document;
9653   d.write(h);
9654   d.focus();
9655   html_ver1("Loaded",w,d);
9656 }
9657 function html_ver1(note,w,d){
9658   if( (v = d.getElementById(htVersionTag)) != null ){
9659     h = v.outerHTML;
9660     u = v.getAttribute('data-url');
9661     t = v.getAttribute('data-time');
9662   }else{
9663     h = 'No version info. in the page';
9664     u = '';
9665     t = 0;
9666   }
9667   showVersion(note,w,v,u,t);
9668 }
9669 function html_ver0(){
9670   html_ver1("Version",window,document);
9671 }
9672 </script>

```

```

9673 <!-- LiveHTML } -->
9674 </span>
9675 </details>
9676 */
9677 /*
9678 <details><summary>Event sharing</summary>
9679 <span id="EventSharingCodeSpan">
9680
9681 <!-- ----- Event sharing // 2020-0925 SatoxITS { -->
9682
9683 <div id="iftestTemplate" class="iftest" hidden="">
9684 <style> .iftestbody{ color:#f22;font-family:Georgia;font-size:10pt;} </style>
9685 <span id="frameBody" class="iftestbody" onclick="frameClick()"><script>
9686   function docadd(txt){
9687     document.body.append(txt);
9688     window.scrollTo(0,100000);
9689   }
9690   function frameClick(){
9691     xy = '(x'+event.x + ' y'+event.y+')';
9692     //docadd('Got Click on #' + event.target.id + ' '+xy+ '\n');
9693     docadd('Got Click on #' + fid.value + ', ' +xy+ '\n');
9694     window.scrollTo(0,100000);
9695     window.parent.postMessage('OnClick: '+xy,'*');
9696   }
9697   function frameMousemove(){
9698     if( false ){
9699       document.body.append('Mousemove on #' + event.target.id + ' '
9700         + 'x'+event.x + ' y'+event.y + '\n');
9701       peerWin = window.frames.frame1;
9702       document.body.append('Send to peer #' + peerWin + ' ' + '\n');
9703       window.scrollTo(0,100000);
9704       peerWin.postMessage('Hi!','*');
9705     }
9706   }
9707   function frameKeydown(){
9708     msg = 'Got Keydown: #' + fid.value + ', (' + event.code + ')';
9709     docadd(msg+'\n');
9710     window.parent.postMessage(msg,'*');
9711   }
9712   function frameOnMessage(){
9713     docadd('Message ' + event.data + '\n');
9714     window.scrollTo(0,100000);
9715   }
9716   if( document.getElementById('Fid') ){
9717     frameBody.id = Fid.value;
9718     h = '';
9719     h += '<' + 'style>*{';
9720     h += 'font-size:10pt;white-space:pre-wrap;';
9721     h += 'font-family:Courier New;';
9722     h += '</' + '/style>';
9723     h += 'I am ' + Fid.value + '\n';
9724     document.write(h);
9725     window.addEventListener('click',frameClick);
9726     window.addEventListener('keydown',frameKeydown);
9727     window.addEventListener('message',frameOnMessage);
9728     window.addEventListener('mousemove',frameMousemove);
9729     window.parent.postMessage('Hi parent, I am ' + Fid.value,'*');
9730   }
9731 }
9732 </script></span></div>
9733
9734 <style>.iframeTest{margin:3px;resize:both;width:370px;height:120px;}</style>
9735 <h2>Inter-window communication</h2>
9736 <note>
9737 frame0 >>> frame1 and frame2<br>
9738 frame1 >>> frame0 and frame2<br>
9739 frame2 >>> frame0 and frame1<br>
9740 </note>
9741 <div id="iframe-test">
9742 <pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
9743 <iframe id="iframe0" title="iframe0" class="iframeTest" style="border:1px;"></iframe>
9744 <iframe id="frame1" title="frame1" class="iframeTest"></iframe>
9745 <iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
9746 </div>
9747
9748 <script id="if0-test-script">
9749   function InterFrameComm_init(){
9750     setupFrames0();
9751     setupFrames12();
9752   }
9753   function setFrameSrcDoc(dst,src){
9754     if( true ){
9755       dst.contentWindow.document.write(src);
9756       // this makes browser wait close, and crash if accumulated !?
9757       // so it should be closed after write
9758       dst.contentWindow.document.close();
9759     }else{
9760       // to be erased before source dump
9761       // but should be set for live snapshot
9762       dst.srccdoc = src;
9763     }
9764   }
9765   function setupFrames0(){
9766     ibody = iframe0.contentWindow.document.body;
9767     iframe0.style.width = "755px";
9768     //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
9769     window.addEventListener('message',messageFromChild);
9770
9771     if0 = '';
9772     if0 += '<' + 'pre style="font-family:Courier New;">';
9773     if0 += '<input id="Fid" value="iframe0">';
9774     if0 += iftestTemplate.innerHTML;
9775     setFrameSrcDoc(iframe0,if0);
9776
9777     function clickOnChild(){
9778       console.log('clickOn #' + this.id);
9779     }
9780     function moveOnChild(){
9781       console.log('moveOn #' + this.id);
9782     }
9783     iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
9784     iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
9785   }
9786   function setupFrames12(){
9787     if1 = '<input id="Fid" value="iframe1">';
9788     if1 += iftestTemplate.innerHTML;
9789     setFrameSrcDoc(iframe1,if1);
9790     //iframe1.name = 'iframe1'; // this seems break contentWindow
9791
9792     if2 = '<input id="Fid" value="iframe2">';
9793     if2 += iftestTemplate.innerHTML;
9794     setFrameSrcDoc(iframe2,if2);
9795
9796     iframe1.addEventListener('message',messageFromChild);

```

```

9797     //iframe1.addEventListener('mouseover',moveOnChild);
9798     iframe2.addEventListener('message',messageFromChild);
9799     //iframe2.addEventListener('mouseover',moveOnChild);
9800     iframe1.contentWindow.postMessage('[parent0] Hi iframe1 -- from parent.','*');
9801     //iframe1.contentWindow.postMessage('Your peer is '+iframe2.contentWindow,'*');
9802     iframe2.contentWindow.postMessage('[parent0] Hi iframe2 -- from parent.','*');
9803     //iframe2.contentWindow.postMessage('Your peer is '+iframe1.contentWindow,'*');
9804   }
9805   function messageFromChild(){
9806     from = null;
9807     forw = null;
9808     if( event.source == iframe0.contentWindow ){
9809       from = '[iframe0]';
9810       forw = 'iframe12';
9811     }else
9812     if( event.source == iframe1.contentWindow ){
9813       from = '[iframe1]';
9814       forw = 'iframe2';
9815     }else
9816     if( event.source == iframe2.contentWindow ){
9817       from = '[iframe2]';
9818       forw = 'iframe1';
9819     }else
9820     {
9821       iframeHost.innerHTML += 'Message [unknown] '
9822       + ' origin' + event.origin
9823       + ' data' + event.data
9824       /* ' from=' + event.source
9825       ;
9826     */
9827     msglog1 = from + event.data + ' -- '
9828     + ' from' + event.source
9829     + ' orig' + event.origin
9830     + ' name' + event.source.name
9831     /* ' port=' + event.ports
9832     /* ' evid=' + event.lastEventId
9833     + '\n'
9834     ;
9835     if( true ){
9836       if( forw == 'iframe1' || forw == 'iframe12' ){
9837         iframe1.contentWindow.postMessage(from+event.data);
9838       }
9839       if( forw == 'iframe2' || forw == 'iframe12' ){
9840         iframe2.contentWindow.postMessage(from+event.data);
9841       }
9842     }
9843     txtadd0(msglog1);
9844   }
9845   function txtadd0(txt){
9846     iframe0.contentWindow.document.body.append(txt);
9847     iframe0.contentWindow.scrollTo(0,100000);
9848   }
9849 }
9850 function es_ShowSelf(){
9851   iframe1.setAttribute('src',document.URL);
9852   iframe2.setAttribute('src',document.URL);
9853 }
9854 </script>
9855 <input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
9856 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
9857 <span id="EventSharingCodeview"></span>
9858 <script id="EventSharingScript">
9859 function es_showHtmlCode(){
9860   showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
9861 }
9862 DestroyEventSharingCodeview = function(){
9863   //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
9864   EventSharingCodeview.innerHTML = "";
9865   iframe0.style = "";
9866   //iframe0.srccode = "erased";
9867   //iframe1.srccode = "erased";
9868   //iframe2.srccode = "erased";
9869   //iframe3.srccode = "erased";
9870 }
9871 </script>
9872 <!-- EventSharing -->
9873 </span>
9874 </details>
9875 */
9876 /**
9877 <!-- ----- "GShell Inside" Notifitaction { -->
9878 <script id="script-gshell-inside">
9879 var notices = 0;
9880 function noticeGShellInside(){
9881   ver = '';
9882   if( ver = document.getElementById('GshVersion') ){
9883     ver = ver.innerHTML;
9884   }
9885   console.log('GJShell Inside (^-^)/*'+ver);
9886   notices += 1;
9887   if( 2 <= notices ){
9888     document.removeEventListener('mousemove',noticeGShellInside);
9889   }
9890 }
9891 document.addEventListener('mousemove',noticeGShellInside);
9892 noticeGShellInside();
9893
9894 const FooterName = 'GshFooter';
9895 function DestroyFooter(){
9896   if( (footer = document.getElementById(FooterName)) != null ){
9897     //footer.parentNode.removeChild(footer);
9898     empty = document.createElement('div');
9899     empty.id = 'GshFooter0';
9900     footer.parentNode.replaceChild(empty,footer);
9901   }
9902 }
9903 function showFooter(){
9904   footer = document.createElement('div');
9905   footer.id = FooterName;
9906   footer.style.backgroundImage = "url("+ITSmoreQR+"")";
9907   //GshFooter0.parentNode.appendChild(footer);
9908   GshFooter0.parentNode.replaceChild(footer,GshFooter0);
9909 }
9910 </script>
9911 <!-- } -->
9912 <!-- border:20px inset #888;
9913 -->
9914 <!--
9915   border:20px inset #888;
9916 -->
9917 //<span id="VirtualDesktopCodeSpan">
9918 /*<details id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
```

```
9921 <!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
9922 <style>
9923 .VirtualSpace {
9924   z-index:0;
9925   width:1280px !important; xheight:720px !important;
9926   width:5120px; height:2880px;
9927   border-width:0px;
9928   xxbackground-color:rgba(32,32,160,0.8);
9929   xxbackground-image:url("WD-WallPaper03.png");
9930   xxbackground-size:100% 100%;
9931   color:#22a;font-family:Georgia;font-size:10pt;
9932   xoverflow:scroll;
9933 }
9934 .VirtualGrid {
9935   z-index:0;
9936   position:absolute;
9937   width:800px; height:500px;
9938   border:1px inset #fff;
9939   color:rgba(192,255,192,0.8);
9940   font-family:Georgia, Courier New;
9941   text-align:right;
9942   vertical-align:middle;
9943   font-size:200px;
9944   text-shadow:4px 4px #ccf;
9945 }
9946 .WD_GridScroll {
9947   z-index:100000;
9948   background-color:rgba(200,200,200,0.1);
9949 }
9950 .VirtualDesktop {
9951   z-index:0;
9952   position:relative;
9953   resize:both !important;
9954   overflow:scroll;
9955   display:block;
9956   min-width:120px !important; min-height:60px !important;
9957   width:800px;
9958   height:500px;
9959   border:10px inset #228;
9960   border-width:30px; border-radius:20px;
9961   background-image:url("WD-WallPaper03.png");
9962   background-size:100% 100%;
9963   color:#22a;font-family:Georgia;font-size:10pt;
9964 }
9965 .comment {
9966   /* overflow:scroll seems to bound childrens' view in the element span
9967   // specifying overflow seems fix the position of the element
9968 */
9969 .VirtualBrowserSpan {
9970   z-index:10;
9971   xxborder:0.5px dashed #fff !important;
9972   border-color:rgba(255,255,255,0.5) !important;
9973   position:relative;
9974   left:100px;
9975   top:100px;
9976   display:block;
9977   resize:both !important;
9978   width:540px;
9979   height:320px;
9980   min-width:40px !important; min-height:20px !important;
9981   max-width:5120px !important; max-height:2880px !important;
9982   background-color:rgba(255,200,255,0.1);
9983   xoverflow:scroll;
9984 }
9985 .xVirtualBrowserLocationBar:focus {
9986   color:#f00;
9987   background-color:rgba(255,128,128,0.2);
9988 }
9989 .xVirtualBrowserLocationBar:active {
9990   color:#f00;
9991   background-color:rgba(128,255,128,0.2);
9992 }
9993 a.VirtualBrowserLocation {
9994   color:#ccc !important;
9995   text-decoration:none !important;
9996 }
9997 a.VirtualBrowserLocation:hover {
9998   color:#fff !important;
9999   text-decoration:underline;
10000 }
10001 .VirtualBrowserLocationBar {
10002   position:absolute;
10003   z-index:100000;
10004   display:block;
10005   width:400px;
10006   height:20px;
10007   padding-left:2px;
10008   line-height:1.1;
10009   vertical-align:middle;
10010   font-size:14px;
10011   color:#fff;
10012   background-color:rgba(128,128,128,0.2);
10013   font-family:Georgia;
10014 }
10015 .VirtualBrowserCommandBar {
10016   position: absolute;
10017   z-index:200000;
10018   xxdisplay:inline;
10019   display:block;
10020   width:60px;
10021   height:20px;
10022   line-height:1.1;
10023   vertical-align:middle;
10024   font-size:14px;
10025   color:#fe4;
10026   background-color:rgba(128,128,128,0.1);
10027   font-family:Georgia;
10028   text-align:left;
10029   left:404px;
10030 }
10031 .VirtualBrowserFrame {
10032   xxposition: relative;
10033   position: absolute;
10034   xxdisplay:inline;
10035   display:block;
10036   z-index:10;
10037   resize:both !important;
10038   width:480px; height:240px;
10039   min-width:60px; min-height:30px;
10040   max-width:5120px; max-height:2880px;
10041   border-radius:6px;
10042   background-color:rgba(255,255,255,0.9);
10043   border-top:20px solid;
10044   border-right:4px solid;
```

```

10045     border-bottom:10px solid;
10046 }
10047 .WinFavicon {
10048     width:16px;
10049     height:16px;
10050     margin:1px;
10051     margin-right:3px;
10052     vertical-align:middle;
10053     background-color:rgba(255,255,255,1.0);
10054 }
10055 .VirtualDesktopMenuBar {
10056     xposition:absolute;
10057     color:#fff;
10058     font-size:7pt;
10059     text-align:right;
10060     padding-right:4px;
10061     background-color:rgba(128,128,128,0.7);
10062 }
10063 .VirtualDesktopCalender {
10064     color:#fff;
10065     font-size:22pt;
10066     text-align:right;
10067     padding-right:4px;
10068     xbackground-color:rgba(255,255,255,0.2);
10069 }
10070 xxxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
10071     display:inline !important; font-size:10pt !important; padding:1px !important;
10072 }
10073 .WD_Config {
10074     display:inline !important;
10075     padding:2px !important;
10076     font-size:10pt !important;
10077     width:60pt !important;
10078     height:12pt !important;
10079     line-height:1.0pt !important;
10080     height:15pt !important;
10081 }
10082 .WD_Button {
10083     display:inline !important;
10084     padding:2px !important;
10085     color:#fff !important;
10086     background-color:#228 !important;
10087     font-size:10pt !important;
10088     width:60pt !important;
10089     height:12pt !important;
10090     line-height:1.0pt !important;
10091     height:16pt !important;
10092     border:2px inset #44a !important;
10093 }
10094 .WD_Href {
10095     display:inline !important;
10096     padding:2px !important;
10097     font-size:9pt !important;
10098     width:120pt !important;
10099     height:12pt !important;
10100     line-height:1.0pt !important;
10101     height:15pt !important;
10102 }
10103
10104 .LiveHtmlCodeviewText {
10105     font-size:10pt;
10106     font-family:Courier New;
10107     xwhite-space:pre;
10108 }
10109
10110 .WD_Panel {
10111     x-index:100 !important;
10112     color:#000 !important;
10113     margin-left:25px !important;
10114     width:800px !important;
10115     padding:4px !important;
10116     border:1px solid #888 !important;
10117     border-radius:6px !important;
10118     background-color:rgba(220,220,220,0.9) !important;
10119     font-size:9pt;
10120     font-family:Courier New;
10121 }
10122 .WD_Help {
10123     font-size:10pt !important;
10124     font-family:Courier New;
10125     line-height:1.2 !important;
10126     color:#000 !important;
10127     width:100% !important;
10128     background-color:rgba(240,240,255,0.8) !important;
10129 }
10130
10131 .WB_Zoom {
10132 }
10133 </style>
10134 <h2>CosmoScreen 0.0.8</h2>
10135 <menu>
10136 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
10137 g ... grid on/off<br>
10138 i ... zoom in<br>
10139 o ... zoom out<br>
10140 s ... save current scope<br>
10141 r ... restore saved scope<br>
10142</span>
10143 </menu>
10144 <div class="WD_Panel" draggable="true">
10145 <p>!-- should be on the frame of the WD -->
10146 Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
10147 < <input id="WD_Width_1" class="WD_Config" type="text">
10148 x <input id="WD_Height_1" class="WD_Config" type="text">
10149 wall-paper: <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaler03.png</a>
10150 </p>
10151 <p>
10152 Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
10153 < <input id="WS_1_Width" class="WD_Config" type="text">
10154 x <input id="WS_1_Height" class="WD_Config" type="text">
10155</p>
10156 <p>
10157 Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
10158 < <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
10159 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
10160 < <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
10161 < <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
10162 </p>
10163 <p>
10164 Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
10165 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
10166 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
10167 shift/wheel for horizontal scroll
10168</p>

```

```

1016<p>
1017Scopexx <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
10171 < <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
10172 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
10173 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
10174</p>
1017< p>
1017Scopeyy <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
10177 < <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
10178 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
10179 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
1018</p>
1018< p>
1018Scopezz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
10183 < <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
10184 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
10185 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
1018</p>
1018< p>
1018Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
1018</p>
1019< p>
1019Overflow <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
10192"scroll" imprisons windows inside the display
1019</p>
1019</div>
10195
1019<div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" contenteditable="true" style="">
1019<div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
1019<i>CosmoScreen 0.0.8</i><span id="VirtualDesktop_1_Clock"></span>
1019</div>
1020<div id="VirtualDesktop_1_Calender" class="VirtualDesktopCalender" >00:00</div>
10201<div align="right"><h1>VirtualSpace 1.</h1></div>
10202<div id="VirtualDesktop_1_Content" class="VirtualSpace">
10203
1020<div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
1020<div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar"></div>
10201<span id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar">Reload</span>
10202<iframe id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""></iframe>
1020</div>
10209
1021<div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
1021<div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
10211<span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar">Reload</span>
10212<iframe id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""></iframe>
10214</div>
10215
1021<div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
1021<div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar"></div>
10211<span id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar">Reload</span>
10212<iframe id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""></iframe>
1022</div>
10221
1022<div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
10222</div>
10224</div>
10225
10226<input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">
10227<span id="VirtualDesktopCodeview"></span>
10228<script id="VirtualDesktopScript">
10229function vd_showHtmlCode(){
10230    codespan = document.getElementById('VirtualDesktopCodeSpan');
10231    showHtmlCode(VirtualDesktopCodeview,codespan);
10232    VirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
10233}
10234DestroyEventSharingCodeview = function(){
10235    VirtualDesktopCodeview.innerHTML = "";
10236}
10237
10238function wdlog(log){
10239    if( GJ_Channel != null ){
10240        GJ_SendMessage('WD '+log);
10241    }
10242    console.log(log);
10243}
10244var topMostWin = 10000;
10245function onEnterWin(e){
10246    t = e.target;
10247    oindex = t.style.zIndex;
10248    //if( oindex == '' ) oindex = 0;
10249    //t.saved_zIndex = oindex;
10250    //t.style.zIndex = 10000;
10251    topMostWin += 1;
10252    t.style.zIndex = topMostWin;
10253    nindex = t.style.zIndex;
10254    wdlog('Enter '+t.id+' #'+oindex+'->'+nindex+'');
10255    e.stopPropagation();
10256    e.preventDefault();
10257}
10258function onClickWin(e){ // can detect click on the thick border? t = e.target;
10259    oindex = t.style.zIndex;
10260    topMostWin += 1;
10261    t.style.zIndex = topMostWin;
10262    nindex = t.style.zIndex;
10263    wdlog('Click '+t.id+' #'+oindex+'->'+nindex+'');
10264    //e.stopPropagation();
10265    //e.preventDefault();
10266}
10267function onLeaveWin(e){
10268    t = e.target;
10269    //oindex = t.style.zIndex;
10270    //nindex = t.saved_zIndex;
10271    //t.style.zIndex = nindex;
10272    //wdlog('Leave '+e.target+' #' +e.target.id+'('+oindex+'->'+nindex+')');
10273    e.stopPropagation();
10274    e.preventDefault();
10275}
10276
10277var WinDragstartX; // event
10278var WinDragstartY;
10279var WindragstartTX; // target
10280var WindragstartTY;
10281
10282function onWinDragstart(e){
10283    WinDragstartX = e.x;
10284    WinDragstartY = e.y;
10285
10286    t = e.target;
10287
10288    //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
10289    //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
10290    if( t.style.left == '' ){
10291        WinDragstartTX = x0 = 0;
10292        t.style.left = '0px';

```

```

10293     }else{
10294         //WinDragstartTX = x0 = Number(t.style.left);
10295         WinDragstartTX = x0 = parseInt(t.style.left);
10296     }
10297     if( t.style.top == '' ){
10298         WinDragstartTY = y0 = 0;
10299         t.style.top = '0px';
10300     }else{
10301         //WinDragstartTY = y0 = Number(t.style.top);
10302         WinDragstartTY = y0 = parseInt(t.style.top);
10303     }
10304     if( true ){ // to be undo
10305         t.wasAtX = WinDragstartTX;
10306         t.wasAtY = WinDragstartTY;
10307     }
10308     wdlog('DragSTA #' + t.id
10309         + ' event(' + e.clientX + ',' + e.clientY + ')'
10310         + ' position=' + t.style.position
10311         + ' style left,top(' + t.style.left + ',' + t.style.top + ')'
10312     );
10313     e.stopPropagation();
10314     //e.preventDefault();
10315     return true;
10316 }
10317 function onWinDragEvent(wh,e,set,dolog){
10318     t = e.target;
10319     dx = e.clientX - WinDragstartX;
10320     dy = e.clientY - WinDragstartY;
10321     nx = WinDragstartX + dx;
10322     ny = WinDragstartY + dy;
10323     log = 'Drag' + wh + '#' + t.id
10324         + ' event0(' + WinDragstartX + ',' + WinDragstartY + ')'
10325         + ' event(' + e.clientX + ',' + e.clientY + ')'
10326         + ' diff(' + dx + ',' + dy + ')'
10327         + ' (' + nx + ',' + ny + ')'
10328         + ' (' + t.style.left + ',' + t.style.top + ')'
10329         + ' wasAt(' + t.wasAtX + ',' + t.wasAtY + ')';
10330 ;
10331     if( e.clientX != 0 || e.clientY != 0 ){
10332         if( set == true ){
10333             //t.style.x = nx + 'px'; // not effective
10334             //t.style.y = ny + 'px'; // not effective
10335             t.style.left = nx + 'px';
10336             t.style.top = ny + 'px';
10337             log += ' Set';
10338         }else{
10339             log += ' NotSet';
10340             if( !dolog ){
10341                 log = '';
10342             }
10343         }
10344     }else{
10345         log += ' What?'; // the type is event start?
10346         if( !dolog ){
10347             log = '';
10348         }
10349     }
10350     if( and(dolog, log != '') ){
10351         wdlog(log);
10352     }
10353     if( true ){
10354         // should be propargeted to parent in FireFox ?
10355         e.stopPropagation();
10356     }
10357     e.preventDefault();
10358     return false;
10359 }
10360 function onWinDrag(e){
10361     return onWinDragEvent('Ing',e,true,false);
10362 }
10363 function onWinDragend(e){
10364     return onWinDragEvent('End',e,false,true);
10365 }
10366 function onWinDragexit(e){
10367     return onWinDragEvent('Exit',e,false,true);
10368 }
10369 function onWinDragover(e){
10370     return onWinDragEvent('Over',e,false,true);
10371 }
10372 function onWinDragenter(e){
10373     return onWinDragEvent('Enter',e,false,true);
10374 }
10375 function onWinDragleave(e){
10376     return onWinDragEvent('Leave',e,false,true);
10377 }
10378 function onWinDragdrop(e){
10379     return onWinDragEvent('Drop',e,false,true);
10380 }
10381 function onFaviconChange(e){
10382     wdlog('--Favicon #' + e.target.id + ' href=' + e.details.href);
10383 }
10384 var savedSuppressGJShell = false;
10385 function stopGJShell(e){
10386     //wdlog('enter Gsh STOP');
10387     savedSuppressGJShell = SuppressGJShell;
10388     SuppressGJShell = true;
10389     e.stopPropagation();
10390     e.preventDefault();
10391 }
10392 function contGJShell(e){
10393     //wdlog('leave Gsh STOP');
10394     SuppressGJShell = savedSuppressGJShell;
10395     e.stopPropagation();
10396     e.preventDefault();
10397 }
10398 function WD_onKeydown(e){
10400     keycode = e.code;
10401     console.log('keydown #' + e.target.id + ' ' + keycode);
10402     if( keycode == 'KeyG' ){
10403         WD_setGrid1(WD_Grid_1);
10404     }else
10405     if( keycode == 'KeyI' ){
10406         WD_doZoomIN();
10407     }else
10408     if( keycode == 'KeyO' ){
10409         WD_doZoomOUT();
10410     }else
10411     if( keycode == 'KeyR' ){
10412         WD_RestoreScope(null);
10413     }else
10414     if( keycode == 'KeyS' ){
10415         WD_SaveScope(null);
10416     }

```

```

10417     e.stopPropagation();
10418     e.preventDefault();
10419 }
10420 function WD_onKeyup(e){
10421     e.stopPropagation();
10422     e.preventDefault();
10423 }
10424 function WD_EventSetup1(){
10425     VirtualDesktop_1.addEventListener('keydown', e => { WD_onKeydown(e); });
10426     VirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onKeydown(e); });
10427     WD_Help_1.addEventListener('keydown', e => { WD_onKeydown(e); });
10428     WD_Help_1.addEventListener('keyup', e => { WD_onKeyup(e); });
10429 }
10430 function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
10431     function VirtualBrowserCommand(e,s,l,cmd,f){
10432         command = cmd.innerHTML;
10433         if( command == "Reload" ){
10434             href_id = e.target.href.id;
10435             d = document.getElementById(href_id);
10436             wdlog('href_tag=' + href_id + '\n elem=' + href_id + '\n href=' + d);
10437             url = d.innerHTML;
10438             wdlog('---- Load href_tag=' + href_id + '\n elem=' + href_id + '\n href=' + d
10439             + '\n url=' + url);
10440             wdlog('---- Load target #' + f.id) + ' with url=' + url;
10441             f.src = url;
10442         }else{
10443             alert('unknown command' + command + ' ' + e.target.id + ', ' + l.id + ', ' + f.id);
10444         }
10445     }
10446     function onKeyDown(e){
10447         if( e.code == 'Enter' ){
10448             e.stopPropagation();
10449             e.preventDefault();
10450         }
10451     }
10452     function onKeyUp(e){
10453         if( e.code == 'Enter' ){
10454             e.stopPropagation();
10455             e.preventDefault();
10456             // should reload immediately ?
10457         }
10458     }
10459     if( false ){
10460         wdlog('start settle VirtualBrowser url=' + u + '\n'
10461             + 'id=' + s.id + '\n'
10462             + 'width=' + s.style.width + '\n'
10463             + 'height=' + s.style.height
10464         );
10465     }
10466     // very important for WordPress ??
10467     s.style.width = f.style.width = 501; // for WordPress ...??
10468     s.style.height = f.style.height = 271; // for WordPress ...??
10469     if( false ){
10470         wdlog('midway settle VirtualBrowser url=' + u + '\n'
10471             + 'id=' + s.id + '\n'
10472             + 'width=' + s.style.width + '\n'
10473             + 'height=' + s.style.height
10474         );
10475     };
10476     s.width = 502; // for WordPress ...??
10477     s.height = 272; // for WordPress ...??
10478     if( false ){
10479         wdlog('midway-2 settle VirtualBrowser url=' + u + '\n'
10480             + 'id=' + s.id + '\n'
10481             + 'span-width=' + s.width + '\n'
10482             + 'span-height=' + s.height
10483         );
10484     };
10485     s.style.width = w + 'px';
10486     s.style.height = h + 'px';
10487     f.style.width = w + 'px';
10488     f.style.height = h + 'px';
10489     //f.style.setProperty('-webkit-transform','scale('+scale+')');
10490     f.style.setProperty('transform','scale('+scale+')');
10491     f.style.setProperty('position','absolute');
10492     f.style.setProperty('left',x+'px');
10493     f.style.setProperty('top',y+'px');
10494     //wdlog('--x1-- u=' + u + ' width s=' + s.style.width + ', f=' + f.style.width);
10495     //wdlog('--x2-- u=' + u + ' width s=' + s.style.width + ', f=' + f.style.width);
10496     f.setAttribute('draggable','true');
10497     f.setAttribute('draggable','false'); // why necessary?
10498     l.setAttribute('draggable','false'); // why necessary?
10499     cmd.setAttribute('draggable','false'); // why necessary?
10500     s.addEventListener('dragstart', e => { onWinDragstart(e); });
10501     s.addEventListener('drag', e => { onWinDrag(e); });
10502     s.addEventListener('exit', e => { onWinDragexit(e); });
10503     s.addEventListener('dragend', e => { onWinDragend(e); });
10504     s.addEventListener('dragexit', e => { onWinDragexit(e); });
10505     s.addEventListener('dragenter', e => { onWinDragenter(e); });
10506     s.addEventListener('dragover', e => { onWinDragover(e); });
10507     s.addEventListener('dragleave', e => { onWinDragleave(e); });
10508     s.addEventListener('drop', e => { onWinDragdrop(e); });
10509     s.addEventListener('mouseenter',e => { onEnterWin(e); });
10510     s.addEventListener('mouseleave',e => { onLeaveWin(e); });
10511
10512     if( false ){
10513         s.style.position = "absolute";
10514         s.style.x = x+'px';
10515         s.style.left = x+'px';
10516         s.style.y = y+'px';
10517         s.style.top = y+'px';
10518     }else{
10519         s.style.setProperty('position','absolute','important');
10520         s.style.setProperty('x',x+'px','important');
10521         s.style.setProperty('left',x+'px','important');
10522         s.style.setProperty('y',y+'px','important');
10523         s.style.setProperty('top',y+'px','important');
10524     }
10525 }
10526
10527 favicon = './favicon.ico';
10528 uvl = u.split('/');
10529 if( 2 <= uvl.length ){
10530     uv2 = uvl[1].split('/');
10531     if( 2 <= uv2.length ){
10532         if( uvl[0] == 'file' ){
10533             //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/');
10534             // + '/favicon.ico';
10535             favicon = './favicon.ico';
10536         }else{
10537             favicon = uvl[0] + '://' + uv2[0] + '/favicon.ico';
10538         }
10539     }
10540 }

```

```

10541 //wdlog("----- favicon-url="+favicon);
10542 href_id = l.id + '_href';
10543 l.innerHTML = '' + u + '</a>';
10545 //l.addEventListener('click', e => { onClickWin(e); });
10546 l.addEventListener('mouseenter', e => { stopGShell(e); });
10547 l.addEventListener('mouseleave', e => { contGShell(e); });
10548 l.addEventListener('keydown', e => { onKeyDown(e); });
10549 l.addEventListener('keyup', e => { onKeyUp(e); });
10550
10551 cmd.href_id = href_id;
10552 wdlog('('1)cmd#'+cmd.id);
10553 wdlog('('2)href id#'+cmd.href_id);
10554 cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
10555
10556 f.style.borderColor = c;
10557 f.src = u;
10558 //f.addEventListener('mouseenter',e => { onEnterWin(e); });
10559 //f.addEventListener('mouseleave',e => { onLeaveWin(e); });
10560
10561 //s.addEventListener('click', e => { onClickWin(e); });
10562 //f.addEventListener('click', e => { wdlog('click wbl'); });
10563 f.addEventListener('mozbrowsericonchange',onFaviconChange);
10564
10565 wdlog('done settle WirtualBrowser url=' + u + '\n'
10566 + 'id=' + s.id + ' '
10567 + 'width=' + s.style.width + ' '
10568 + 'height=' + s.style.height + ' '
10569 + 'cmd=' + cmd.id
10570 );
10571 );
10572 }
10573
10574 function WD_EventSetup2(){
10575 dt = WirtualDesktop_1;
10576 dt.style.width = "800px";
10577 dt.style.height = "500px";
10578 dt.addEventListener('dragstart',e => { onWinDragstart(e); });
10579 dt.addEventListener('drag', e => { onWinDrag(e); });
10580 dt.addEventListener('exit', e => { onWinDragexit(e); });
10581 }
10582
10583 function GronClick(){
10584 WD_SaveScope(null); // should be push
10585 t = event.target;
10586 x = t.getAttribute('data-leftx');
10587 y = t.getAttribute('data-topy');
10588 zoom = WD_Zoom_1_XY.value;
10589 x *= zoom;
10590 y *= zoom;
10591 WD_doScrollXY(event,x,y);
10592 //alert('scroll #' + t.id + ' x=' + x + ', y=' + y);
10593 }
10594 function WD_SetGrid(e){
10595 t = e.target;
10596 WD_SetGrid1(t);
10597 }
10598 function WD_SetGrid1(t){
10599 //ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10600 ds = WirtualDesktop_1_GridPlane;
10601 if( t.value == 'GridOn' ){
10602 for( col = 0; col < 16; col++ ){
10603 for( row = 0; row < 16; row++ ){
10604 g1 = document.createElement('span');
10605 g1.setAttribute('class','VirtualGrid');
10606 leftx = col * 800;
10607 topy = row * 500;
10608 gid = col + '.' + row
10609 label = '<+' + span +
10610 + 'id="' + gid + '" '+'class="WD_GridScroll" ' +
10611 + 'contenteditable="false" onclick="GronClick()" ' +
10612 + 'data-leftx=' + leftx + ' ' + 'data-topy=' + topy + ' ' +
10613 + '>' +
10614 + gid + '<+/' + span + '>';
10615 console.log('grid ' + label);
10616 g1.innerHTML = label;
10617 g1.position = 'relative';
10618 g1.leftx = leftx;
10619 g1.topy = topy;
10620 g1.style.left = g1.leftx + 'px';
10621 g1.style.top = g1.topy + 'px';
10622 if( col % 2 == row % 2 ){
10623 g1.style.backgroundColor = 'rgba(255,255,255,0.3)';
10624 }
10625 ds.appendChild(g1);
10626 }
10627 }
10628 t.value = 'GridOff';
10629 }else{
10630 ds.innerHTML = '';
10631 t.value = 'GridOn';
10632 }
10633 }
10634
10635 var sav_scrollLeft;
10636 var sav_scrollTop;
10637 var sav_nscale;
10638 function WD_SaveScope(e){
10639 sav_scrollLeft = WD_Left_1.value;
10640 sav_scrollTop = WD_Top_1.value;
10641 sav_nscale = WD_Zoom_1_XY.value;
10642 //console.log('saved zoom=' + sav_nscale + ', ' + sav_nscale);
10643 }
10644 function WD_RestoreScope(e){
10645 WD_Zoom_1_XY.value = sav_nscale;
10646 WD_doZoom();
10647
10648 WD_Left_1.value = sav_scrollLeft;
10649 WD_Top_1.value = sav_scrollTop;
10650 WD_doScroll(null);
10651 }
10652 function ignoreEvent(e){
10653 e.stopPropagation();
10654 //e.preventDefault();
10655 }
10656 function zoomMag(){
10657 return WD_Zoom_1_MAG.value;
10658 }
10659 function WD_EventSetup3(){
10660 WD_Grid_1.addEventListener('click', e => { WD_SetGrid(e); });
10661 WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
10662 WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
10663 WD_Width_1.value = dt.style.width;
10664 WD_Width_1.addEventListener('keydown', ignoreEvent);

```

```

10665 WD_Width_1.addEventListener('keyup',ignoreEvent);
10666 WD_Height_1.value = dt.style.height;
10667 WD_Height_1.addEventListener('keydown',ignoreEvent);
10668 WD_Height_1.addEventListener('keyup',ignoreEvent);
10669 WD_Zoom_1_MAG.addEventListener('keydown',ignoreEvent);
10670 WD_Zoom_1_MAG.addEventListener('keyup',ignoreEvent);
10671 }
10672
10673 function escale1(e,oscale,nscale){
10674   e.style.setProperty('transform','scale('+nscale+')');
10675   rscale = oscale / nscale;
10676   w = parseInt(e.style.width);
10677   h = parseInt(e.style.height);
10678   w = w * rscale; //oscale/nscale;
10679   h = h * rscale; //oscale/nscale;
10680   e.style.width = w + 'px';
10681   e.style.height = h + 'px';
10682 }
10683 function scaleWD(ds,oscale,nscale){
10684   if( true ){
10685     escale1(WirtualBrowser_1,oscale,nscale);
10686     escale1(WirtualBrowser_1_Location,oscale,nscale);
10687     escale1(WirtualBrowser_1_Command,oscale,nscale);
10688     escale1(WirtualBrowser_1_Frame,oscale,nscale);
10689
10690     escale1(WirtualBrowser_2,oscale,nscale);
10691     escale1(WirtualBrowser_2_Location,oscale,nscale);
10692     escale1(WirtualBrowser_2_Command,oscale,nscale);
10693     escale1(WirtualBrowser_2_Frame,oscale,nscale);
10694
10695     escale1(WirtualBrowser_3,oscale,nscale);
10696     escale1(WirtualBrowser_3_Location,oscale,nscale);
10697     escale1(WirtualBrowser_3_Command,oscale,nscale);
10698     escale1(WirtualBrowser_3_Frame,oscale,nscale);
10699   }
1070 }
1070 function WD_doZoom(){
10702   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
10703   oscale = WD_Zoom_1_XY.value; // hidden value for current zoom
10704   nscale = WD_Zoom_1_XY.value;
10705   ds.style.zoom = nscale;
10706   WD_Zoom_1_XY.value = ds.style.zoom;
10707   WD_Zoom_1_XY.value = ds.style.zoom;
10708   scaleWD(ds,oscale,nscale);
10709 }
10710 function WD_EventSetup4(){
10711   WD_Zoom_1.addEventListener('click',WD_doZoom);
10712   WD_Zoom_1_XY.addEventListener('keydown',ignoreEvent);
10713   WD_Zoom_1_XY.addEventListener('keyup',ignoreEvent);
10714 }
10715
10716 function WD_doZoomOUT(){
10717   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
10718   oscale = WD_Zoom_1_XY.value;
10719   if( oscale == 0 || oscale == '' ){
10720     oscale = 1;
10721   }
10722   nscale = oscale / zoomMag();
10723   ds.style.zoom = nscale;
10724   WD_Zoom_1_XY.value = ds.style.zoom;
10725   WD_Zoom_1_XY.value = ds.style.zoom;
10726   scaleWD(ds,oscale,nscale);
10727 }
10728 function WD_doZoomIN(){
10729   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
10730   oscale = WD_Zoom_1_XY.value;
10731   if( oscale == 0 || oscale == '' ){
10732     oscale = 1;
10733   }
10734   nscale = oscale * zoomMag();
10735   if( 4 < nscale ){
10736     alert('maybe too large, zoom=' + nscale);
10737     return;
10738   }
10739   ds.style.zoom = nscale;
10740   WD_Zoom_1_XY.value = ds.style.zoom;
10741   WD_Zoom_1_XY.value = ds.style.zoom;
10742   scaleWD(ds,oscale,nscale);
10743 }
10744 function WD_EventSetup5(){
10745   WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
10746   WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
10747 }
10748
10749 function WD_doResize(e){
10750   dt = WirtualDesktop_1;
10751   dt.style.width = WD_Width_1.value;
10752   dt.style.height = WD_Height_1.value;
10753   WD_Width_1.value = dt.style.width;
10754   WD_Height_1.value = dt.style.height;
10755 }
10756 WD_Resize_1.addEventListener('click',e => { WD_doResize(e); });
10757
10758
10759 function WD_doRSResize(e){
10760   //alert('Resize Space');
10761   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
10762   ds.style.width = WS_1_Width.value;
10763   ds.style.height = WS_1_Height.value;
10764   WS_1_Width.value = ds.style.width;
10765   WS_1_Height.value = ds.style.height;
10766 }
10767 function WD_EventSetup6(){
10768   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
10769   ds.style.width = '5120px';
10770   ds.style.height = '2880px';
10771   WS_1_Width.value = ds.style.width;
10772   WS_1_Height.value = ds.style.height;
10773   WS_1_Width.addEventListener('keydown', ignoreEvent);
10774   WS_1_Width.addEventListener('keyup', ignoreEvent);
10775   WS_1_Height.addEventListener('keydown', ignoreEvent);
10776   WS_1_Height.addEventListener('keyup', ignoreEvent);
10777   WS_Resize_1.addEventListener('click',e => { WD_doRSResize(e); });
10778 }
10779
10780 function WD_doScrollXY(e,sleft,stop){
10781   dt = WirtualDesktop_1;
10782   dt.scrollLeft = sleft;
10783   dt.scrollTop = stop;
10784   WD_Left_1.value = dt.scrollLeft;
10785   WD_Top_1.value = dt.scrollTop;
10786   console.log('--Scroll #' + dt.id + ' (' + sleft + ',' + stop + ')');
10787 }
10788 function WD_doScroll(e){

```

```

10789 //dt = VirtualDesktop_1_Content;
10790 dt = VirtualDesktop_1;
10791 sleft = parseInt(WD_Left_1.value);
10792 stop = parseInt(WD_Top_1.value);
10793 dt.scrollLeft = sleft;
10794 dt.scrollTop = stop;
10795 WD_Left_1.value = dt.scrollLeft;
10796 WD_Top_1.value = dt.scrollTop;
10797 console.log('--Scroll #' + dt.id + ' (' + sleft + ',' + stop + ')');
10798 }
10799 function showScrollPosition(){
10800 if( false )
10801 console.log(
10802   'wstop=' + VirtualDesktop_1.style.top + ',' +
10803   'wsx=' + VirtualDesktop_1.style.y + ',' +
10804   'wsz=' + VirtualDesktop_1.scrollTop + ',' +
10805   'wdtop=' + VirtualDesktop_1_Content.style.top + ',' +
10806   'wdx=' + VirtualDesktop_1_Content.style.y + ',' +
10807   'wds=' + VirtualDesktop_1_Content.scrollTop + ',' +
10808 );
10809 WD_Left_1.value = VirtualDesktop_1.scrollLeft;
10810 WD_Top_1.value = VirtualDesktop_1.scrollTop;
10811 }
10812 function WD_EventSetup7(){
10813 WD_Scroll_1.addEventListener('click',e => { WD_doScroll(e); });
10814 WD_Left_1.addEventListener('keydown',ignoreEvent);
10815 WD_Left_1.addEventListener('keyup',ignoreEvent);
10816 WD_Top_1.addEventListener('keydown',ignoreEvent);
10817 WD_Top_1.addEventListener('keyup',ignoreEvent);
10818 }
10819 function WD_EventSetup8(){
10820 VirtualDesktop_1.addEventListener('scroll',showScrollPosition);
10821 VirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
10822 }
10823
10824 if( false ){
10825 w = 1000 + 'px';
10826 dt.style.width = w;
10827 dt.style.height = "300px";
10828 dt.style.resize = 'both';
10829 dt.style.borderWidth = 50 + 'px';
10830 dt.style.borderRadius = 25 + 'px';
10831 console.log('----- #' + dt.id + ' style=' + dt.style);
10832 console.log('----- #' + dt.id + ' width=' + dt.style.width);
10833 console.log('----- #' + dt.id + ' left=' + dt.style.left);
10834 console.log('----- #' + dt.id + ' border=' + dt.style.border);
10835 }
10836 function onDTResize(e){
10837 dt = e.target;
10838 h = parseInt(dt.style.height);
10839 dt.style.borderWidth = (h * 0.075) + 'px';
10840 console.log('----- borderWidgh=' + dt.style.borderWidth);
10841 }
10842
10843 VirtualDesktopDetails.addEventListener('toggle',VirtualDesktop_init);
10844 function VirtualDesktop_init(){
10845 if( !VirtualDesktopDetails.open ){
10846 return;
10847 }
10848 //GJ_Join();
10849 VirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
10850 //console.log('----- #' + dt.id +
10851 // +' borderWidth=' + dt.style.getProperty('border-width'));
10852
10853 settleWin(
10854   VirtualBrowser_1,
10855   VirtualBrowser_1_Location,
10856   VirtualBrowser_1_Command,
10857   VirtualBrowser_1_Frame,
10858   document.URL,
10859   500,280,50,20,'#262',1.0);
10860 settleWin(
10861   VirtualBrowser_2,
10862   VirtualBrowser_2_Location,
10863   VirtualBrowser_2_Command,
10864   VirtualBrowser_2_Frame,
10865   'https://its-more.jp/ja_jp/',
10866   500,280,150,100,'#448',1.0);
10867 settleWin(
10868   VirtualBrowser_3,
10869   VirtualBrowser_3_Location,
10870   VirtualBrowser_3_Command,
10871   VirtualBrowser_3_Frame,
10872   '/.../gshell/gsh.go.html',
10873   // 'http://gshell.org/gshell/gsh.go.html',
10874   'https://golang.org',
10875   500,280,250,180,'#444',1.0);
10876 //1000,720,0,0,#444,0.125;
10877 //1200,720,-100,-50,#444,0.4);
10878 function WD_ClockUpdate(e){
10879   VirtualDesktop_1_Clock.innerHTML = DateShort();
10880   VirtualDesktop_1_Calendar.innerHTML = DateHourMin();
10881 }
10882 window.setInterval(WD_ClockUpdate,500);
10883
10884 WD_EventSetup1();
10885 WD_EventSetup2();
10886 WD_EventSetup3();
10887 WD_EventSetup4();
10888 WD_EventSetup5();
10889 WD_EventSetup6();
10890 WD_EventSetup7();
10891 WD_EventSetup8();
10892 }
10893 //VirtualDesktop_init();
10894
10895 Destroy_VirtualDesktop = function(){
10896   VirtualDesktop_1.style = "";
10897
10898   VirtualBrowser_1.removeAttribute('style');
10899   VirtualBrowser_1_Location.innerHTML = '';
10900   VirtualBrowser_1_Frame.removeAttribute('src');
10901   VirtualBrowser_1_Frame.removeAttribute('style');
10902   VirtualBrowser_1_Frame.style="";
10903
10904   VirtualBrowser_2.removeAttribute('style');
10905   VirtualBrowser_2_Location.innerHTML = '';
10906   VirtualBrowser_2_Frame.removeAttribute('src');
10907   VirtualBrowser_2_Frame.style="";
10908
10909   VirtualBrowser_3.removeAttribute('style');
10910   VirtualBrowser_3_Location.innerHTML = '';
10911   VirtualBrowser_3_Frame.removeAttribute('src');
10912   VirtualBrowser_3_Frame.style="";

```

```

10913     GJFactory_1.style = "";
10914     iframe0.style = "";
10915     VirtualDesktop_1.style = "";
10916 }
10917 }
10918
10919</script>
10920<!-- VirtualDesktop } -->
10921</details>
10922*/ //</span>
10923
10924//<!-- ===== Work { ===== -->
10925//<span id="SBSidebar_WorkCodeSpan">
10926/*
10927<details><summary>SBSidebar</summary>
10928<!-- ----- SBSidebar // 2020-0928 SatoxITS { -->
10929<h2>SBSidebar</h2>
10930<input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
10931<input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
10932<input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="showCode">
10933<span id="SBSidebar_WorkCodeView"></span>
10934<script id="SBSidebar_WorkScript">
10935    function SBSidebar_openWorkCodeView(){
10936        function SBSidebar_showWorkCode(){
10937            showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
10938        }
10939        SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
10940    }
10941 SBSidebar_openWorkCodeView(); // should be invoked by an event
10942
10943 console.log('-- SbSlider // 2020-1006-01 SatoxITS --');
10944 function SetSidebar(){
10945     sidebar = document.getElementById('secondary');
10946     // console.log('primary=' + primary + ' secondary=' + sidebar + ' main=' + main + ' ');
10947     wrap = sidebar.parentNode;
10948     console.log('-- SbSlider parent is ' + wrap, '#'+wrap.id' .'+wrap.class);
10949//wrap = wrap.parentNode;
10950//console.log('-- SbSlider parent is '+wrap, '#'+wrap.id' .'+wrap.class);
10951//wrap = wrap.parentNode;
10952//console.log('-- SbSlider parent is '+wrap, '#'+wrap.id' .'+wrap.class);
10953//nsb = sidebar.cloneNode();
10954     nsb = sidebar;
10955     nsb.style.width = '100';
10956     slider = document.createElement('div');
10957     slider.id = 'SbSlider';
10958     slider.appendChild(nsb);
10959     slider.setAttribute('class','SbSlider');
10960     nsb.style.position = 'relative';
10961     slider.style.position = 'fixed';
10962     slider.style.display = 'block'; // inline';
10963     slider.style.zIndex = 10000;
10964     // nsb.style.zIndex = 20000;
10965     nsb.style.position = 'absolute';
10966     nsb.style.minWidth = '80px';
10967     nsb.style.left = '0px';
10968     nsb.style.top = '0px';
10969
10970     w = window.innerWidth;
10971     console.log('SliderWidth '+w+: ',(w/3)+'px');
10972     if( w < 640 ){
10973         slider.style.setProperty('width',(w/3) + 'px','important');
10974     }
10975     main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
10976
10977     slider.style.resize = "both";
10978     slider.draggable = "true";
10979     wrap.appendChild(slider);
10980     console.log('-- added SbSlider');
10981     //nsb.addEventListener('scroll',SbScrolled);
10982
10983     buttons = document.createElement('div');
10984     buttons.id = 'NaviButtons';
10985     buttons.setAttribute('class','NaviButtons');
10986     buttons.align = "center";
10987     buttons.innerHTML += '<+'+><a href="#TopOfPost" draggable="true">TOP<+'+><+'></a><+'></p>';
10988     buttons.innerHTML += '<+'+><a href="#EndOfPost" draggable="true">END<+'></p>';
10989     page.appendChild(buttons);
10990     buttons.style.position = 'fixed';
10991     buttons.style.zIndex = 3000;
10992     buttons.style.width = '100%';
10993     buttons.style.top = '320px';
10994     buttons.style.left = parseInt(w) * 1.0 + 'px';
10995     console.log('-- SbSlider installed (^-^)/ SatoxITS');
10996 }
10997//window.addEventListener('load',SetSidebar); // after load
10998DestroyNaviButtons = function(){
10999     nb = document.getElementById('NaviButtons');
11000     if( nb != null ){
11001         nb.parentNode.removeChild(NaviButtons);
11002     }
11003 }
11004</script>
11005
11006// 2020-1006 its-more.jp-blog-60000-style.css
11007<!-- {
11008<style>
11009#NaviButtons {
11010    position:fixed;
11011    display:block;
11012    width:100%;
11013    top:100px;
11014    left:10px;
11015    z-index:30000;
11016    font-size:10pt;
11017    color:#2ff !important;
11018    text-align:center;
11019    background-color:rgba(230,230,230,0.01);
11020}
11021#NaviButtons a {
11022    color:#2a2 !important;
11023    font-size:20px;
11024    text-align:center;
11025    text-decoration:2px 2px #8ff;
11026    resize:both;
11027    padding:6px;
11028    margin:10px;
11029    border:1px solid #288 !important;
11030    border-radius:3px;
11031    background-color:rgba(160,160,160,0.05);
11032}
11033#SbSlider {
11034    overflow:auto;
11035    resize:both !important;
11036    overflow-y:hidden !important;

```

```

11037 height:100px !important;
11038 display:inline !important;
11039 position:fixed !important;
11040 left:0px;
11041 top:0px;
11042 xxwidth:180px;
11043 width:24px;
11044 min-width:80px;
11045 height:100% !important;
11046 background-color:rgba(100,100,200,0.1);
11047}
11048#secondary {
11049 position:fixed;
11050 left:0px;
11051 top:0px;
11052 xxxx-index:60000;
11053 scroll-behavior: overflow !important;
11054 xxxxxxxxxxxxxxxxxxxxxxxxwidth:18% !important;
11055 padding-left:pt;
11056 color:#fff;
11057 font-size:0.5em;
11058 background-color:rgba(64,160,64,0.6) !important;
11059 white-space:nowrap;
11060}
11061#secondary a {
11062 color:#fff !important;
11063 text-decoration:underline !important;
11064}
11065#primary {
11066 position:relative;
11067 width:75% !important;
11068 left:25% !important;
11069}
11070#main {
11071 position:relative;
11072 width:75% !important;
11073 left:25% !important;
11074}
11075#site-navigation {
11076 position:relative;
11077 left:120px;
11078}
11079#adswsc_countertext {
11080 color:#4169e1;
11081 font-size:16pt !important;
11082 xxfont-size:10% !important;
11083 font-weight:bold;
11084}
11085#nowTime {
11086 color:#aofffa0;
11087 font-size:16pt !important;
11088 xxfont-size:10% !important;
11089 font-weight:bold;
11090 text-shadow:1px 1px #fff;
11091}
11092.navigation-top {
11093 color:#22a !important;
11094 border:0px;
11095 background-color:rgba(220,220,220,0.1);
11096}
11097
11098.visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
11099 display: block;
11100 xxwidth: lem;
11101 xxoverflow: auto;
11102 xxheight: lem;
11103}
11104.invisible-scrollbar ::-webkit-scrollbar {
11105 xxdisplay: none;
11106 width:1px !important;
11107 height:1px !important;
11108}
11109.mostly-customized-scrollbar ::-webkit-scrollbar {
11110 width: 2px;
11111 height: 2px;
11112 xxbackground-color: #aaa; xxx:or add it to the track;
11113}
11114.mostly-customized-scrollbar ::-webkit-scrollbar-thumb {
11115 background: #000;
11116}
11117</style>
11118} -->
11119
11120</details>
11121<!--_SBSidebar_WorkCodeSpan } -->
11122</-- //</span>
11124//<!-- ===== Work } ===== -->
11125
11126
11127//<!-- ===== Work { ===== -->
11128<span id="Affiliate_WorkCodeSpan">
11129/*
1113<div id="AffSet" class="AffSet" draggable="true">
11131 <iframe id="aff_0" src="/gshell/gsh.go.html" class="AffTarget"></iframe>
11132 <iframe id="aff_1" src="https://golang.org" class="AffTarget"></iframe>
11133 <iframe id="aff_2" src="https://drafts.csswg.org/" class="AffTarget"></iframe>
11134 <iframe id="aff_3" src="https://html.spec.whatwg.org/dev/" class="AffTarget"></iframe>
11135 <iframe id="aff_4" src="https://www.wikipedia.org" class="AffTarget"></iframe>
11136 <iframe id="aff_5" src="https://www.bing.com/translator" class="AffTarget"></iframe>
11137</div>
11138<details><summary>Affiliate // 2020-10-10 SatoxITS { -->
11139<!-- ----- Affiliate // 2020-10-10 SatoxITS { -->
11140<h2>Supportive Affiliate</h2>
11141<style>
11142.AffSet {
11143 position:fixed;
11144 max-width:2560px;
11145 max-height:1440px;
11146 width:270px;
11147 left:75px;
11148 height:100px;
11149 resize:both;
11150 xoverflow:visible;
11151 overflow:scroll;
11152 xleft:-10px;
11153 top:0px;
11154 xleft:0px;
11155 xxalign:right;
11156 z-index:10000;
11157 display:block;
11158 xborder:1px inset #44a;
11159 border:1px inset rgba(255,255,255,0.1);
11160 background-color:rgba(255,255,255,0.1);

```

```

11161}
11162.AffSet:hover {
11163    z-index:10000000;
11164    border:2px inset #f00;
11165    background-color:rgba(80,80,255,0.3);
11166    xbackground-color:#eee;
11167}
11168.AffTarget {
11169    max-width:2560px;
11170    max-height:1440px;
11171    z-index:1000000;
11172    display:block;
11173    xposition:fixed;
11174    xposition:absolute;
11175    position:relative;
11176    //left:300px;
11177    xresize:both;
11178    padding:0px;
11179    width:600px;
11180    height:400px;
11181    max-height:800px !important;
11182    margin-top:0%;
11183    margin-left:0%;
11184    margin-right:0% !important;
11185    border:16px inset #ccc;
11186    transform:scale(0.5);
11187    background-color:rgba(255,255,255,0.8);
11188    xxalign:right;
11189}
11190.AffTarget:hover {
11191    border:16px inset #bbf;
11192}
11193</style>
11194<input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11195<input id="Affiliate_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11196<input id="Affiliate_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11197<span id="Affiliate_WorkCodeView"></span>
11198<script id="Affiliate_WorkScript">
11199function Affiliate_openWorkCodeView(){
11200    function Affiliate_showWorkCode(){
11201        showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
11202    }
11203    Affiliate_WorkCodeView.addEventListener('click',Affiliate_showWorkCode);
11204}
11205Affiliate_openWorkCodeView(); // should be invoked by an event
11206
11207//<iframe id="aff_8" xsrc="https://stackoverflow.com/tags" class="AffTarget"></iframe>
11208//<iframe id="aff_9" xsrc="https://developer.mozilla.org/en-US/docs/Web" class="AffTarget"></iframe>
11209function affsetup(){
11210    parent = document.documentElement;
11211    parent.appendChild(AffSet);
11212    AffSet.style.top = '0px';
11213    AffSet.style.left = (window.innerWidth - 280) + 'px';
11214
11215    var off = 100;
11216    zoom = 0.5;
11217    ozoom = 0.3;
11218    leftx = window.innerWidth - 300;
11219    left = leftx + 'px';
11220    left = -130 + 'px';
11221    console.log('aff-init window.innerWidth=' + window.innerWidth);
11222    w = 1000;
11223    h = 560;
11224
11225    //parent.appendChild(aff_0);
11226    aff_0.style.width = zoom*w+'px';
11227    aff_0.style.height = zoom*h+'px';
11228    aff_0.style.left = left;
11229    //aff_0.style.top = off+'px'; off += ozoom*h;
11230    aff_0.draggable = 'true';
11231
11232    //parent.appendChild(aff_1);
11233    aff_1.style.width = zoom*w+'px';
11234    aff_1.style.height = zoom*h+'px';
11235    aff_1.style.left = left;
11236    //aff_1.style.top = off+'px'; off += ozoom*h;
11237    aff_1.style.top = '-150px';
11238    aff_1.draggable = 'true';
11239
11240    //parent.appendChild(aff_2);
11241    aff_2.style.width = zoom*w+'px';
11242    aff_2.style.height = zoom*h+'px';
11243    aff_2.style.left = left;
11244    //aff_2.style.top = off+'px'; off += ozoom*h;
11245    aff_2.style.top = '-300px';
11246    aff_2.draggable = 'true';
11247
11248    //parent.appendChild(aff_3);
11249    aff_3.style.transform = 'scale(0.25)';
11250    aff_3.style.width = 2*zoom*w+'px';
11251    aff_3.style.height = 2*zoom*h+'px';
11252    aff_3.style.border = '32px inset #ccc';
11253    //aff_3.style.left = -390 + 'px'; //left*2;
11254    //aff_3.style.left = (leftx - 265) + 'px';
11255    aff_3.style.left = -395 + 'px';
11256    //aff_3.style.top = (-155+off)+'px'; off += ozoom*h;
11257    aff_3.style.top = '-600px';
11258    aff_3.draggable = 'true';
11259
11260    //parent.appendChild(aff_4);
11261    aff_4.style.width = zoom*w+'px';
11262    aff_4.style.height = zoom*h+'px';
11263    aff_4.style.left = left;
11264    //aff_4.style.top = off+'px'; off += ozoom*h;
11265    aff_4.style.top = '-900px';
11266    aff_4.draggable = 'true';
11267
11268    //parent.appendChild(aff_5);
11269    aff_5.style.transform = 'scale(0.300)';
11270    aff_5.style.width = zoom*(w*1.67)+'px';
11271    aff_5.style.height = zoom*(h*1.67)+'px';
11272    aff_5.style.border = '25px inset #ccc';
11273    aff_5.style.left = -308+'px';
11274    //aff_5.style.left = (-175+leftx)+'px';
11275    //aff_5.style.top = (-95+off)+'px'; off += ozoom*h;
11276    //aff_5.style.left = '0px';
11277    //aff_5.style.top = '0px';
11278    aff_5.style.top = '-1150px';
11279    //aff_5.style.align = 'right';
11280    aff_5.draggable = 'true';
11281}
11282function affresize(){
11283    AffSet.style.left = (window.innerWidth - 280) + 'px';
11284    leftx = window.innerWidth - 400;

```

```
11285     left = leftx + 'px';
11286     console.log('aff-resize window.innerWidth=' + window.innerWidth);
11287 }
11288 window.addEventListener('resize', affresize);
11289 //document.addEventListener('resize', affresize);
11290 //gsh.addEventListener('resize', affresize);
11291 </script>
11292 </details>
11293 <!-- Affiliate_WorkCodeSpan } -->
11294 /* /</span>
11295 //<!-- ===== Work } ===== -->
11296
11297
11298 //<!-- ===== Work { ===== -->
11299 //<span id="OriginalSource_WorkCodeSpan">
11300 /*
11301 <details open=""><summary>Original Source</summary>
11302 <!-- ===== OriginalSource // 2020-1009 SatoITS { -->
11303 <h2>Original Source of GShell</h2>
11304 <input id="OriginalSource_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11305 <input id="OriginalSource_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11306 <input id="OriginalSource_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11307 <span id="OriginalSourceTextElement"></span>
11308 <span id="OriginalSource_WorkCodeScript">
11309 <script id="OriginalSource_WorkScript">
11310 <function OriginalSource_OpenWorkCodeView(){
11311     function OriginalSource_showWorkCode(){
11312         //showHtmlCode(OriginalSource_WorkCodeView,OriginalSource_WorkCodeSpan);
11313         //OriginalSourceTextElement = OriginalSourceNode;
11314         //console.log('src3=\n'+OriginalSourceNode.outerHTML);
11315         showHtmlCode(OriginalSource_WorkCodeView,OriginalSourceNode);
11316     }
11317     OriginalSource_WorkCodeViewOpen.addEventListener('click',OriginalSource_showWorkCode);
11318 }
11319 //OriginalSourceNode = document.documentElement.cloneNode();
11320 //OriginalSourceNode = gsh.cloneNode(true); //=====
11321 //console.log('src0=\n'+document.documentElement.outerHTML);
11322 //console.log('src1=\n'+gsh.outerHTML);
11323 //console.log('src2=\n'+OriginalSourceNode.innerHTML);
11324 OriginalSource_OpenWorkCodeView(); // should be invoked by an event
11325 //showNodeAsHtmlSource(OriginalSource_WorkCodeView,OriginalSourceNode);
11326 function SaveOriginalNode(){
11327     if( false ){
11328         m0 = performance.memory;
11329         mu0 = m0.usedJSHeapSize;
11330         console.log('-- heap bef clone: '
11331             +m0.usedJSHeapSize+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
11332     }
11333     OriginalSourceNode = gsh.cloneNode(true);
11334     if( false ){
11335         m1 = performance.memory;
11336         mul = m1.usedJSHeapSize;
11337         mu = mul - mu0;
11338         //alert('-- clone: used heap '+mu0+' -> '+mul+' = '+mu+' bytes');
11339         console.log('-- heap aft clone: '
11340             +m1.usedJSHeapSize+'/'+m1.totalHeapSize+'/'+m1.jsHeapSizeLimit);
11341     //OriginalSourceNode = document.documentElement.cloneNode(true);
11342 }
11343 }
11344
11345 function Gsh_setupPage(){
11346     GshSetImages();
11347     //Indexer_afterLoaded();
11348     //GShell_initKeyCommands();
11349     //GJConsole_initConsole();
11350     GJConsole_initFactory();
11351     GJLink_init();
11352     InterFrameComm_init();
11353     Gshell_initTopbar();
11354     //VirtualDesktop_init();
11355     Banner_init();
11356     affsetup();
11357     window.setInterval(ShowResourceUsage,1000);
11358 }
11359 function OnLoad(){
11360     SaveOriginalNode();
11361     Gsh_setupPage();
11362 }
11363 document.addEventListener('load',Gsh_setupPage);
11364 </script>
11365 </details>
11366 <!-- OriginalSource_WorkCodeSpan } -->
11367 /* /</span>
11368 //<!-- ===== Work } ===== -->
11369
11370
11371
11372 //</div>
11373 //<br><script>OnLoad();</script></span></html>
11374
```