

```

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--0.6.2--2020-10-07--SatoxITS</span>
7  <title>GShell-0.6.2 by SatoxITS</title>
8
9  <div id="GshSidebar" class="SbSidebar"><div id="GshIndexer"></div></div>
10 <div id="GshMain">
11 <header id="GshBanner" height="100px" onclick="shiftBG();">
12 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.6.2 // 2020-10-07 // SatoxITS</note></div>
13 </header>
14
15 <!-- ----- Work { ----- ---->
16 </span id="Indexer_WorkCodeSpan">
17 /*
18 <details open><summary>Indexer</summary>
19 <!-- ----- Indexer // 2020-1007 SatoxITS { -->
20 <h2>Indexer</h2>
21 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
22 <input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
23 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
24 <span id="Indexer_WorkCodeView"></span>
25 </details>
26 <style id="SidebarIndex">
27 #gsh {
28   display:block;
29   overflow:scroll !important;
30 }
31 #GshMain {
32   position:relative;
33   width:80% !important;
34   left:19.5% !important;
35 }
36 #GshSidebar {
37   z-index:0;
38   position:relative !important;
39   overflow:auto;
40   resize:both !important;
41   xxoverflow-y:hidden !important;
42   height:100px !important;
43   xxxdisplay:inline !important;
44   left:0px;
45   top:0px;
46   width:19.5%;
47   min-width:80px;
48   height:100% !important;
49   color:#f00;
50   xxbackground-color:rgba(64,64,64,0.5);
51 }
52 #GshIndexer {
53   z-index:0;
54   position:relative;
55   height:100%;
56   left:0px;
57   top:0px;
58   scroll-behavior: overflow !important;
59   padding-left:4pt;
60   color:#f00;
61   font-size:0.5em;
62   background-color:rgba(64,160,64,0.6) !important;
63   white-space:nowrap;
64 }
65 .IndexLine {
66   font-size:8pt !important;
67   font-family:Georgia;
68   display:block;
69   color:#fff;
70   padding-right:4pt;
71 }
72 .IndexLine:hover {
73   color:#228;
74   background-color:#fff;
75   xtext-decoration:underline !important;
76 }
77 </style>
78
79 <script id="Indexer_WorkScript">
80 function Indexer_openWorkCodeView(){
81   function Indexer_showWorkCode(){
82     showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
83   }
84   Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
85 }
86 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
87 Indexer_openWorkCodeView();
88
89 var iserno = 0;
90 var GeneratedId = 0;
91 function generateIndex(ni,e,chv,nch,ht){
92   // https://developer.mozilla.org/en-US/docs/Web/API/Element
93   c = '';
94   if( e.classList != null ){
95     c = e.classList.value;
96   }
97   console.log('-- <'+e.nodeName+'> #' +e.id+' .'+c+' '+e.attributes);
98   if( e.nodeName == '#text' ){ return ''; }
99   if( e.nodeName == '#comment' ){ return ''; }
100  if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
101    id = e.innerHTML;
102    GeneratedId += 1;
103    eid = 'GeneratedId-'+GeneratedId;
104    e.id = eid;
105  }else
106  if( e.nodeName == 'SUMMARY' ){
107    id = e.innerHTML;
108    GeneratedId += 1;
109    eid = 'GeneratedId-'+GeneratedId;
110    e.id = eid;
111  }else
112  if( and(e.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxentry-content' ) ){
113    console.log('-- DIV entry-content begin');
114    id = e.innerHTML;
115    GeneratedId += 1;
116    eid = 'GeneratedId-'+GeneratedId;
117    e.id = eid;
118    console.log('-- DIV entry-content end hash-child=',e.hasChildNodes());
119  }else
120  if( e.id == '' || e.id == 'undefined' ){
121    return '';
122  }else{
123    id = '#' +e.id;
124    eid = e.id;

```

```

125 }
126 iserno += 1;
127 ht = '<+div id="GeneratedEref '+iserno+' class="IndexLine" href="'+eid+'>'
128 + iserno+' '+ni+' '+e.nodeName + ' ' + id;
129 if( e.id == '' || e.id == 'undefined' ){ return ht + '<+div>'; }
130 if( !e.hasChildNodes() ){ return ht + '<+div>'; }
131 chv = e.childNodes;
132 nch = e.childNodes.length;
133 if( chv != null ){ nch = chv.length; }
134 ht += ' ('+nch+' ) + '<+div>';
135 for( let i = 0; i < chv.length; i++){
136     sec = ni+'.'+i;
137     if( ni == '' ){ sec = i; }
138     ht += generateIndex(sec,chv[i],null,0);
139 }
140 return ht;
141 }
142 function onClickIndex(e){
143     tid = e.target.id;
144     tge = document.getElementById(tid);
145     eid = tge.getAttribute('href');
146     rx = tge.getBoundingClientRect().left.toFixed(0)
147     ry = tge.getBoundingClientRect().top.toFixed(0)
148     if( false ){
149         alert('index clicked mouse(x='+e.x+', y='+e.y+')'
150             + '\ntid=#'+ tid + ' rx='+rx + ',ry='+ry
151             + '\neid= ' + eid + '\n'
152             + '\nhtml='+ tge.outerHTML);
153     }
154     ee = document.getElementById(eid);
155     sx = 'NaN';
156     sy = ee.getBoundingClientRect().top;
157     console.log('sx'+sx+',sy'+sy);
158     ee.scrollIntoView()
159     window.scrollTo(sx,sy)
160     //window.scrollTo({left:'Non',top:sy,behavior:'smooth'});
161 }
162 function Indexer_afterLoaded(){
163     sideindex = document.getElementById('GshIndexer');
164     ht = '<+h3>GShell Index<+h3>';
165     ht += generateIndex("",document.getElementById('gsh'),null,0,'');
166     if( (pri = document.getElementById('primary')) != null ){
167         ht += generateIndex("",pri,null,0,'');
168     }
169     ht += '<+br>';
170     ht += '<+br>';
171     ht += '<+br>';
172     ht += '<+br>';
173     sideindex.innerHTML = ht;
174     sideindex.addEventListener('click',onClickIndex);
175 }
176 if( (pri = document.getElementById('primary')) != null ){
177     console.log('-- Seems in WordPress');
178     pri.style.zIndex = 2000;
179 }
180 GshSidebar.style.setProperty('position','relative','important');
181 GshSidebar.style.top = '-1400px';
182 //GshSidebar.style.setProperty('position','absolute','important');
183 //GshSidebar.style.top = '0px';
184
185 GshSidebar.style.setProperty('width','200px','important');
186 GshSidebar.style.setProperty('overflow','scroll','important');
187 GshSidebar.style.resize = 'both';
188
189 GshSidebar.style.left = '-100px';
190 GshIndexer.style.left = '100px';
191 GshIndexer.style.height = '1400px';
192 gsh.appendChild(GshSidebar); // change parent
193 }else{
194     console.log('-- Seems not in WordPress');
195     GshSidebar.style.setProperty('position','fixed','important');
196 }
197 }
198 //document.addEventListener('load',Indexer_afterLoaded);
199
200 DestroyIndexBar = function(){
201     sideindex = document.getElementById('GshIndexer');
202     sideindex.innerHTML = "";
203 }
204 </script>
205
206 <!-- Indexer_WorkCodeSpan -->
207 * //</span>
208 <!-- ----- Work } ----- -->
209
210 /*
211 <h2>GShell // a General purpose Shell built on the top of Golang</h2>
212 <p>
213 <note>
214 It is a shell for myself, by myself, of myself. --SatoxITS(^-^ )
215 <a href="gsh-0.6.1.go.html">prev.</a>
216 </note>
217 </p>
218 <div id="GJFactory_x"></div>
219
220 <div>
221 <span id="GshMenu">
222 <span class="GshMenu" id="GshMenuEdit" onclick="html_edit();">Edit</span>
223 <span class="GshMenu" id="GshMenuSave" onclick="html_save();">Save</span>
224 <span class="GshMenu" id="GshMenuLoad" onclick="html_load();">Load</span>
225 <span class="GshMenu" id="GshMenuVers" onclick="html_ver0();">Vers</span>
226 <span id="gsh-WinId" onclick="win_jump('0.1');">0</span>
227 <span class="GshMenu" id="gsh-menu-exit" onclick="html_close();"></span>
228 <span class="GshMenu" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
229 <span class="GshMenu" id="gsh-menu-stop" onclick="html_stop(this,true);">Stop</span>
230 <span class="GshMenu" id="gsh-menu-fold" onclick="html_fold(this);">Unfold</span>
231 <span class="GshMenu" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
232 <span class="GshMenu" id="gsh-menu-sign" onclick="html_sign(this);" style="">Source</span>
233 <!-- | <span id="gsh-menu-pure" onclick="html_pure(this);">Pure</span> -->
234 </span>
235 </div>
236 </div>
237 *//
238
239 /*
240 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
241 <h3>Fun to create a shell</h3>
242 <p>For a programmer, it must be far easy and fun to create his own simple shell
243 rightly fitting to his favor and necessities, than learning existing shells with
244 complex full features that he never use.
245 I, as one of programmers, am writing this tiny shell for my own real needs,
246 totally from scratch, with fun.
247 </p><p>
248 For a programmer, it is fun to learn new computer languages. For long years before

```

```

249 writing this software, I had been specialized to C and early HTML2 :-).
250 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
251 on demand as a novice of these, with fun.
252 </p></p>
253 This single file "gsh.go", that is executable by Go, contains all of the code written
254 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
255 HTML file that works as the viewer of the code of itself, and as the "home page" of
256 this software.
257 </p></p>
258 Because this HTML file is a Go program, you may run it as a real shell program
259 on your computer.
260 But you must be aware that this program is written under situation like above.
261 Needless to say, there is no warranty for this program in any means.
262 </p>
263 <address>Aug 2020, SatoxITS (sato@its-more.jp)</address>
264 </details>
265 */
266 /*
267 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
268 </p>
269 <h3>Cross-browser communication</h3>
270 <p>
271 ... to be written ...
272 </p>
273 <h3>Vi compatible command line editor</h3>
274 <p>
275 The command line of GShell can be edited with commands compatible with
276 <a href="https://www.washington.edu/computing/unix/vi.html"><b>vi</b></a>.
277 As in vi, you can enter <i><b>command mode</b></i> by <b>ESC</b> key,
278 then move around in the history by <b><code>j k / ? n N</code></b>,
279 or within the current line by <b><code>l h f w b 0 $ %</code></b> or so.
280 </p>
281 </details>
282 */
283 /*
284 <details id="gsh-gindex">
285 <summary>Index</summary><div class="gsh-src">
286 Documents
287 <span class="gsh-link" onclick="jumpto_JavaScriptView();">Command summary</span>
288 Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
289 Package structures
290 <a href="#import">import</a>
291 <a href="#struct">struct</a>
292 Main functions
293 <a href="#comexpansion">str-expansion</a> // macro processor
294 <a href="#finder">finder</a> // builtin find + du
295 <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
296 <a href="#plugin">plugin</a> // plugin commands
297 <a href="#ex-commands">system</a> // external commands
298 <a href="#builtin">builtin</a> // builtin commands
299 <a href="#network">network</a> // socket handler
300 <a href="#remote-sh">remote-sh</a> // remote shell
301 <a href="#redirect">redirect</a> // StdIn/Out redirection
302 <a href="#history">history</a> // command history
303 <a href="#rusage">rusage</a> // resource usage
304 <a href="#encode">encode</a> // encode / decode
305 <a href="#IME">IME</a> // command line IME
306 <a href="#getline">getline</a> // line editor
307 <a href="#scanf">scanf</a> // string decomposer
308 <a href="#interpreter">interpreter</a> // command interpreter
309 <a href="#main">main</a>
310 </span>
311 JavaScript part
312 <a href="#script-src-view" class="gsh-link" onclick="jumpto_JavaScriptView();">Source</a>
313 <a href="#gsh-data-frame" class="gsh-link" onclick="jumpto_DataView();">Builtin data</a>
314 CSS part
315 <a href="#style-src-view" class="gsh-link" onclick="jumpto_StyleView();">Source</a>
316 References
317 <a href="#" class="gsh-link" onclick="jumpto_WholeView();">Internal</a>
318 <a href="#gsh-reference" class="gsh-link" onclick="jumpto_ReferenceView();">External</a>
319 Whole parts
320 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Source</a>
321 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Download</a>
322 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Dump</a>
323 </div>
324 </details>
325 */
326 /*
327 <details id="gsh-gocode">
328 <summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
329 // gsh - Go lang based Shell
330 // (c) 2020 ITS more Co., Ltd.
331 // 2020-0807 created by SatoxITS (sato@its-more.jp)
332
333 package main // gsh main
334
335 // <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
336 import (
337     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
338     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
339     "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
340     "sort" // <a href="https://golang.org/pkg/sort/">sort</a>
341     "time" // <a href="https://golang.org/pkg/time/">t.time</a>
342     "bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
343     "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
344     "os" // <a href="https://golang.org/pkg/os/">os</a>
345     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
346     "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
347     "net" // <a href="https://golang.org/pkg/net/">net</a>
348     "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
349     "html" // <a href="https://golang.org/pkg/html/">html</a>
350     "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
351     "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
352     "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
353     "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
354     "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
355     // "gshdata" // gshell's logo and source code
356     "hash/crc32" // <a href="https://golang.org/pkg/unicode/hash/crc32/">crc32</a>
357     "golang.org/x/net/websocket"
358 )
359
360 // // 2020-0906 added,
361 // // <a href="https://golang.org/cmd/cgo/">CGo</a>
362 // #include "poll.h" // <poll.h> // </poll.h> to be closed as HTML tag -p
363 // typedef struct { struct pollfd fdv[8]; } pollFdv;
364 // int pollx(pollFdv *fdv, int nfd, int timeout){
365 //     return poll(fdv->fdv,nfds,timeout);
366 // }
367 import "C"
368
369 // // 2020-0906 added,
370 func CFPollIn1(fp*os.File, timeoutUs int)(ready uintptr){
371     var fdv = C.pollFdv{}
372     var nfd = 1

```

```

373     var timeout = timeoutUs/1000
374
375     fdv.fdv[0].fd = C.int(fp.Fd())
376     fdv.fdv[0].events = C.POLLIN
377     if( 0 < EventRecvFd ){
378         fdv.fdv[1].fd = C.int(EventRecvFd)
379         fdv.fdv[1].events = C.POLLIN
380         nfds += 1
381     }
382     r := C.pollx(&fdv,C.int(nfds),C.int(timeout))
383     if( r <= 0 ){
384         return 0
385     }
386     if (int(fdv.fdv[1].revents) & int(C.POLLIN)) != 0 {
387         //fprintf(stderr,"--De-- got Event\n");
388         return uintptr(EventFdOffset + fdv.fdv[1].fd)
389     }
390     if (int(fdv.fdv[0].revents) & int(C.POLLIN)) != 0 {
391         return uintptr(NormalFdOffset + fdv.fdv[0].fd)
392     }
393     return 0
394 }
395
396 const (
397     NAME = "gsh"
398     VERSION = "0.6.2"
399     DATE = "2020-10-07"
400     AUTHOR = "SatoxITS(^-^)"
401 )
402 var (
403     GSH_HOME = ".gsh" // under home directory
404     GSH_PORT = 9999
405     MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
406     PROMPT = ">"
407     LINESIZE = (8*1024)
408     PATHSEP = ":" // should be ";" in Windows
409     DIRSEP = "/" // canbe \ in Windows
410 )
411
412 // -xX logging control
413 // --A-- all
414 // --I-- info.
415 // --D-- debug
416 // --T-- time and resource usage
417 // --W-- warning
418 // --E-- error
419 // --F-- fatal error
420 // --Xn- network
421
422 // <a name="struct">Structures</a>
423 type GCommandHistory struct {
424     StartAt time.Time // command line execution started at
425     EndAt time.Time // command line execution ended at
426     ResCode int // exit code of (external command)
427     CmdError error // error string
428     OutData *os.File // output of the command
429     FoundFile []string // output - result of ufind
430     Rusagev [2]syscall.Rusage // Resource consumption, CPU time or so
431     CmdId int // maybe with identified with arguments or impact
432     // redirection commands should not be the CmdId
433     WorkDir string // working directory at start
434     WorkDirX int // index in ChdirHistory
435     CmdLine string // command line
436 }
437 type GChdirHistory struct {
438     Dir string
439     MovedAt time.Time
440     CmdIndex int
441 }
442 type CmdMode struct {
443     Background bool
444 }
445 type Event struct {
446     when time.Time
447     event int
448     evarg int64
449     CmdIndex int
450 }
451 var CmdIndex int
452 var Events []Event
453 type PluginInfo struct {
454     Spec *plugin.Plugin
455     Addr plugin.Symbol
456     Name string // maybe relative
457     Path string // this is in Plugin but hidden
458 }
459 type GServer struct {
460     host string
461     port string
462 }
463
464 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
465 const ( // SumType
466     SUM_ITEMS = 0x000001 // items count
467     SUM_SIZE = 0x000002 // data length (simply added)
468     SUM_SIZEHASH = 0x000004 // data length (hashed sequence)
469     SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
470     // also envelope attributes like time stamp can be a part of digest
471     // hashed value of sizes or mod-date of files will be useful to detect changes
472
473     SUM_WORDS = 0x000010 // word count is a kind of digest
474     SUM_LINES = 0x000020 // line count is a kind of digest
475     SUM_SUM64 = 0x000040 // simple add of bytes, useful for human too
476
477     SUM_SUM32_BITS = 0x000100 // the number of true bits
478     SUM_SUM32_2BYTE = 0x000200 // 16bits words
479     SUM_SUM32_4BYTE = 0x000400 // 32bits words
480     SUM_SUM32_8BYTE = 0x000800 // 64bits words
481
482     SUM_SUM16_BSD = 0x001000 // UNIXsum -sum -bsd
483     SUM_SUM16_SYSV = 0x002000 // UNIXsum -sum -sysv
484     SUM_UNIXIFILE = 0x004000
485     SUM_CRCIEEE = 0x008000
486 )
487 type CheckSum struct {
488     Files int64 // the number of files (or data)
489     Size int64 // content size
490     Words int64 // word count
491     Lines int64 // line count
492     SumType int
493     Sum64 uint64
494     Crc32Table crc32.Table
495     Crc32Val uint32
496     Sum16 int

```

```

497 Ctime      time.Time
498 Atime      time.Time
499 Mtime      time.Time
500 Start      time.Time
501 Done       time.Time
502 RusageAtStart [2]syscall.Rusage
503 RusageAtEnd [2]syscall.Rusage
504 }
505 type ValueStack []string
506 type GshContext struct {
507     StartDir      string // the current directory at the start
508     GetLine       string // gsh-getline command as a input line editor
509     ChdirHistory  []GchdirHistory // the 1st entry is wd at the start
510     gshPA         syscall.ProcAttr
511     CommandHistory []GCommandHistory
512     CmdCurrent    GCommandHistory
513     Background    bool
514     BackgroundJobs []int
515     LastRusage    syscall.Rusage
516     GshHomeDir    string
517     TerminalId    int
518     CmdTrace      bool // should be [map]
519     CmdTime       bool // should be [map]
520     PluginFuncs  []PluginInfo
521     iValues       []string
522     iDelimiter    string // field sepearater of print out
523     iFormat       string // default print format (of integer)
524     iValStack     ValueStack
525     LastServer    GServer
526     RSERV        string // [gsh://]host[:port]
527     RWD          string // remote (target, there) working directory
528     lastChecksum  CheckSum
529 }
530
531 func nsleep(ns time.Duration){
532     time.Sleep(ns)
533 }
534 func usleep(ns time.Duration){
535     nsleep(ns*1000)
536 }
537 func msleep(ns time.Duration){
538     nsleep(ns*1000000)
539 }
540 func sleep(ns time.Duration){
541     nsleep(ns*1000000000)
542 }
543
544 func strBegins(str, pat string)(bool){
545     if len(pat) <= len(str){
546         yes := str[0:len(pat)] == pat
547         //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat, yes)
548         return yes
549     }
550     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
551     return false
552 }
553 func isin(what string, list []string) bool {
554     for _, v := range list {
555         if v == what {
556             return true
557         }
558     }
559     return false
560 }
561 func isinX(what string,list[]string)(int){
562     for i,v := range list {
563         if v == what {
564             return i
565         }
566     }
567     return -1
568 }
569
570 func env(opts []string) {
571     env := os.Environ()
572     if isin("-s", opts){
573         sort.Slice(env, func(i,j int) bool {
574             return env[i] < env[j]
575         })
576     }
577     for _, v := range env {
578         fmt.Printf("%v\n",v)
579     }
580 }
581
582 // - rewriting should be context dependent
583 // - should postpone until the real point of evaluation
584 // - should rewrite only known notation of symbol
585 func scanInt(str string)(val int,leng int){
586     leng = -1
587     for i,ch := range str {
588         if '0' <= ch && ch <= '9' {
589             leng = i+1
590         }else{
591             break
592         }
593     }
594     if 0 < leng {
595         ival,_ := strconv.Atoi(str[0:leng])
596         return ival,leng
597     }else{
598         return 0,0
599     }
600 }
601 func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
602     if len(str[i+1:]) == 0 {
603         return 0,rstr
604     }
605     hi := 0
606     histlen := len(gshCtx.CommandHistory)
607     if str[i+1] == '!' {
608         hi = histlen - 1
609         leng = 1
610     }else{
611         hi,leng = scanInt(str[i+1:])
612         if leng == 0 {
613             return 0,rstr
614         }
615         if hi < 0 {
616             hi = histlen + hi
617         }
618     }
619     if 0 <= hi && hi < histlen {
620         var ext byte

```

```

621     if l < len(str[i+leng:]) {
622         ext = str[i+leng:][1]
623     }
624     //fmt.Printf("--D-- %v(%)\n",str[i+leng:],str[i+leng])
625     if ext == 'f' {
626         leng += 1
627         xlist := []string{}
628         list := gshCtx.CommandHistory[hi].FoundFile
629         for _,v := range list {
630             //list[i] = escapeWhiteSP(v)
631             xlist = append(xlist,escapeWhiteSP(v))
632         }
633         //rstr += strings.Join(list," ")
634         rstr += strings.Join(xlist," ")
635     }else
636     if ext == 'e' || ext == 'd' {
637         // !N% .. workdir at the start of the command
638         leng += 1
639         rstr += gshCtx.CommandHistory[hi].WorkDir
640     }else{
641         rstr += gshCtx.CommandHistory[hi].CmdLine
642     }
643     }else{
644         leng = 0
645     }
646     return leng,rstr
647 }
648 func escapeWhiteSP(str string)(string){
649     if len(str) == 0 {
650         return "\\z" // empty, to be ignored
651     }
652     rstr := ""
653     for _,ch := range str {
654         switch ch {
655             case '\\': rstr += "\\\\"
656             case ' ': rstr += "\\s"
657             case '\t': rstr += "\\t"
658             case '\r': rstr += "\\r"
659             case '\n': rstr += "\\n"
660             default: rstr += string(ch)
661         }
662     }
663     return rstr
664 }
665 func unescapeWhiteSP(str string)(string){ // strip original escapes
666     rstr := ""
667     for i := 0; i < len(str); i++ {
668         ch := str[i]
669         if ch == '\\' {
670             if i+1 < len(str) {
671                 switch str[i+1] {
672                     case 'z':
673                         continue;
674                 }
675             }
676         }
677         rstr += string(ch)
678     }
679     return rstr
680 }
681 func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
682     ustrv := []string{}
683     for _,v := range strv {
684         ustrv = append(ustrv,unescapeWhiteSP(v))
685     }
686     return ustrv
687 }
688
689 // <a name="comexpansion">str-expansion</a>
690 // - this should be a macro processor
691 func strsubst(gshCtx *GshContext,str string,histonly bool) string {
692     rbuff := []byte{}
693     if false {
694         //@@U Unicode should be cared as a character
695         return str
696     }
697     //rstr := ""
698     inEsc = 0 // escape characer mode
699     for i := 0; i < len(str); i++ {
700         //fmt.Printf("--D--Subst %v:%v\n",i,str[i])
701         ch := str[i]
702         if inEsc == 0 {
703             if ch == '!' {
704                 //leng,xrstr := substHistory(gshCtx,str,i,rstr)
705                 leng,rs := substHistory(gshCtx,str,i,"")
706                 if 0 < leng {
707                     //_,rs := substHistory(gshCtx,str,i,"")
708                     rbuff = append(rbuff,[]byte(rs)...)
709                     i += leng
710                     //rstr = xrstr
711                     continue
712                 }
713             }
714             switch ch {
715                 case '\\': inEsc = '\\'; continue
716                 case '%': inEsc = '%'; continue
717                 case '$':
718                 }
719             }
720             switch inEsc {
721                 case '\\':
722                     switch ch {
723                         case '\\': ch = '\\'
724                         case 's': ch = ' '
725                         case 't': ch = '\t'
726                         case 'r': ch = '\r'
727                         case 'n': ch = '\n'
728                         case 'z': inEsc = 0; continue // empty, to be ignored
729                     }
730                 inEsc = 0
731                 case '%':
732                     switch {
733                         case ch == '%': ch = '%'
734                         case ch == 'T':
735                             //rstr = rstr + time.Now().Format(time.Stamp)
736                             rs := time.Now().Format(time.Stamp)
737                             rbuff = append(rbuff,[]byte(rs)...)
738                             inEsc = 0
739                             continue;
740                         default:
741                             // postpone the interpretation
742                             //rstr = rstr + "%" + string(ch)
743                             rbuff = append(rbuff,ch)
744                             inEsc = 0

```

```

745         continue;
746     }
747     inEsc = 0
748 }
749 //rstr = rstr + string(ch)
750 rbuff = append(rbuff,ch)
751 }
752 //fmt.Printf("--D--subst(%s)(%s)\n",str,string(rbuff))
753 return string(rbuff)
754 //return rstr
755 }
756 func showFileInfo(path string, opts []string) {
757     if isin("-l",opts) || isin("-ls",opts) {
758         fi, err := os.Stat(path)
759         if err != nil {
760             fmt.Printf("----- ((%v))",err)
761         }else{
762             mod := fi.ModTime()
763             date := mod.Format(time.Stamp)
764             fmt.Printf("%v %v %s ",fi.Mode(),fi.Size(),date)
765         }
766     }
767     fmt.Printf("%s",path)
768     if isin("-sp",opts) {
769         fmt.Printf(" ")
770     }else
771     if ! isin("-n",opts) {
772         fmt.Printf("\n")
773     }
774 }
775 func userHomeDir()(string,bool){
776     /*
777     homedir,_ = os.UserHomeDir() // not implemented in older Golang
778     */
779     homedir,found := os.LookupEnv("HOME")
780     //fmt.Printf("--I-- HOME=%v(%v)\n",homedir,found)
781     if !found {
782         return "/tmp",found
783     }
784     return homedir,found
785 }
786
787 func toFullpath(path string) (fullpath string) {
788     if path[0] == '/' {
789         return path
790     }
791     pathv := strings.Split(path,DIRSEP)
792     switch {
793     case pathv[0] == ".":
794         pathv[0],_ = os.Getwd()
795     case pathv[0] == "..": // all ones should be interpreted
796         cwd,_ := os.Getwd()
797         ppathv := strings.Split(cwd,DIRSEP)
798         pathv[0] = strings.Join(ppathv,DIRSEP)
799     case pathv[0] == "-":
800         pathv[0],_ = userHomeDir()
801     default:
802         cwd,_ := os.Getwd()
803         pathv[0] = cwd + DIRSEP + pathv[0]
804     }
805     return strings.Join(pathv,DIRSEP)
806 }
807
808 func IsRegFile(path string)(bool){
809     fi, err := os.Stat(path)
810     if err == nil {
811         fm := fi.Mode()
812         return fm.IsRegular();
813     }
814     return false
815 }
816
817 // <a name="encode">Encode / Decode</a>
818 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
819 func (gshCtx *GshContext)Enc(argv[]string){
820     file := os.Stdin
821     buff := make([]byte,LINESIZE)
822     li := 0
823     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
824     for li = 0; ; li++ {
825         count, err := file.Read(buff)
826         if count <= 0 {
827             break
828         }
829         if err != nil {
830             break
831         }
832         encoder.Write(buff[0:count])
833     }
834     encoder.Close()
835 }
836 func (gshCtx *GshContext)Dec(argv[]string){
837     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
838     li := 0
839     buff := make([]byte,LINESIZE)
840     for li = 0; ; li++ {
841         count, err := decoder.Read(buff)
842         if count <= 0 {
843             break
844         }
845         if err != nil {
846             break
847         }
848         os.Stdout.Write(buff[0:count])
849     }
850 }
851 // lnspl [N] [-crlf][-C \\\]
852 func (gshCtx *GshContext)SplitLine(argv[]string){
853     strRep := isin("-str",argv) // "..."
854     reader := bufio.NewReaderSize(os.Stdin,64*1024)
855     ni := 0
856     toi := 0
857     for ni = 0; ; ni++ {
858         line, err := reader.ReadString('\n')
859         if len(line) <= 0 {
860             if err != nil {
861                 fmt.Fprintf(os.Stderr,"--I-- lnspl %d to %d (%v)\n",ni,toi,err)
862                 break
863             }
864         }
865         off := 0
866         ilen := len(line)
867         remlen := len(line)
868         if strRep { os.Stdout.Write([]byte("\n")) }

```

```

869     for oi := 0; 0 < remlen; oi++ {
870         olen := remlen
871         addnl := false
872         if 72 < olen {
873             olen = 72
874             addnl = true
875         }
876         fmt.Fprintf(os.Stderr, "--D-- write %d [%d.%d] %d %d/%d/%d\n",
877             toi, ni, oi, off, olen, remlen, ilen)
878         toi += 1
879         os.Stdout.Write([]byte(line[0:olen]))
880         if addnl {
881             if strRep {
882                 os.Stdout.Write([]byte("\n\n"))
883             } else {
884                 //os.Stdout.Write([]byte("\r\n"))
885                 os.Stdout.Write([]byte("\\"))
886                 os.Stdout.Write([]byte("\n"))
887             }
888             line = line[olen:]
889             off += olen
890             remlen -= olen
891         }
892     }
893     if strRep { os.Stdout.Write([]byte("\n")) }
894 }
895 fmt.Fprintf(os.Stderr, "--I-- lnspp %d to %d\n", ni, toi)
896 }
897
898 // CRC32 <a href="http://golang.jp/pkg/hash-crc32">crc32</a>
899 // 1 0000 0100 1100 0001 0001 1101 1011 0111
900 var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
901 var CRC32IEEE uint32 = uint32(0xEDB88320)
902 func byteCRC32add(crc uint32, str []byte, len uint64)(uint32){
903     var oi uint64
904     for oi = 0; oi < len; oi++ {
905         var oct = str[oi]
906         for bi := 0; bi < 8; bi++ {
907             //fprintf(stderr, "--CRC32 %d %X (%d.%d)\n", crc, oct, oi, bi)
908             ovf1 := (crc & 0x80000000) != 0
909             ovf2 := (oct & 0x80) != 0
910             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
911             oct <<= 1
912             crc <<= 1
913             if ovf { crc ^= CRC32UNIX }
914         }
915     }
916     //fprintf(stderr, "--CRC32 return %d %d\n", crc, len)
917     return crc;
918 }
919 func byteCRC32end(crc uint32, len uint64)(uint32){
920     var slen = make([]byte, 4)
921     var li = 0
922     for li = 0; li < 4; {
923         slen[li] = byte(len)
924         li += 1
925         len >>= 8
926         if( len == 0 ){
927             break
928         }
929     }
930     crc = byteCRC32add(crc, slen, uint64(li))
931     crc ^= 0xFFFFFFFF
932     return crc
933 }
934 func strCRC32(str string, len uint64)(crc uint32){
935     crc = byteCRC32add(0, []byte(str), len)
936     crc = byteCRC32end(crc, len)
937     //fprintf(stderr, "--CRC32 %d %d\n", crc, len)
938     return crc
939 }
940 func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
941     var slen = make([]byte, 4)
942     var li = 0
943     for li = 0; li < 4; {
944         slen[li] = byte(len & 0xFF)
945         li += 1
946         len >>= 8
947         if( len == 0 ){
948             break
949         }
950     }
951     crc = crc32.Update(crc, table, slen)
952     crc ^= 0xFFFFFFFF
953     return crc
954 }
955
956 func (gsh *GshContext)xChecksum(path string, argv []string, sum *Checksum)(int64){
957     if !isin("-type/f", argv) && !isRegFile(path){
958         return 0
959     }
960     if !isin("-type/d", argv) && isRegFile(path){
961         return 0
962     }
963     file, err := os.OpenFile(path, os.O_RDONLY, 0)
964     if err != nil {
965         fmt.Printf("--E-- cksum %v (%v)\n", path, err)
966         return -1
967     }
968     defer file.Close()
969     if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n", path, argv) }
970
971     bi := 0
972     var buff = make([]byte, 32*1024)
973     var total int64 = 0
974     var initTime = time.Time{}
975     if sum.Start == initTime {
976         sum.Start = time.Now()
977     }
978     for bi = 0; ; bi++ {
979         count, err := file.Read(buff)
980         if count <= 0 || err != nil {
981             break
982         }
983         if (sum.SumType & SUM_SUM64) != 0 {
984             s := sum.Sum64
985             for _, c := range buff[0:count] {
986                 s += uint64(c)
987             }
988             sum.Sum64 = s
989         }
990         if (sum.SumType & SUM_UNIXFILE) != 0 {
991             sum.Crc32Val = byteCRC32add(sum.Crc32Val, buff, uint64(count))
992         }

```



```

993     if (sum.SumType & SUM_CRCIEEB) != 0 {
994         sum.Crc32Val = crc32.Update(sum.Crc32Val, &sum.Crc32Table, buff[0:count])
995     }
996     // <a href="https://en.wikipedia.org/wiki/BSD_checksum">BSD checksum</a>
997     if (sum.SumType & SUM_SUM16_BSD) != 0 {
998         s := sum.Sum16
999         for _, c := range buff[0:count] {
1000             s = (s >> 1) + ((s & 1) << 15)
1001             s += int(c)
1002             s &= 0xFFFF
1003             //fmt.Printf("BSDsum: %d[%d] %d\n", sum.Size+int64(i), i, s)
1004         }
1005         sum.Sum16 = s
1006     }
1007     if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1008         for bj := 0; bj < count; bj++ {
1009             sum.Sum16 += int(buff[bj])
1010         }
1011     }
1012     total += int64(count)
1013 }
1014 sum.Done = time.Now()
1015 sum.Files += 1
1016 sum.Size += total
1017 if !isin("-s", argv) {
1018     fmt.Printf("%v ", total)
1019 }
1020 return 0
1021 }
1022
1023 // <a name="grep">grep</a>
1024 // "lines", "lin" or "lmp" for "(text) line processor" or "scanner"
1025 // a*,lab,c, ... sequential combination of patterns
1026 // what "LINE" is should be definable
1027 // generic line-by-line processing
1028 // grep [-v]
1029 // cat -n -v
1030 // uniq [-c]
1031 // tail -f
1032 // sed s/x/y/ or awk
1033 // grep with line count like wc
1034 // rewrite contents if specified
1035 func (gsh*GshContext)xGrep(path string, rexpv[]string)(int){
1036     file, err := os.OpenFile(path, os.O_RDONLY, 0)
1037     if err != nil {
1038         fmt.Printf("--E-- grep %v (%v)\n", path, err)
1039         return -1
1040     }
1041     defer file.Close()
1042     if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n", path, rexpv) }
1043     //reader := bufio.NewReaderSize(file, LINESIZE)
1044     reader := bufio.NewReaderSize(file, 80)
1045     li := 0
1046     found := 0
1047     for li = 0; ; li++ {
1048         line, err := reader.ReadString('\n')
1049         if len(line) <= 0 {
1050             break
1051         }
1052         if 150 < len(line) {
1053             // maybe binary
1054             break;
1055         }
1056         if err != nil {
1057             break
1058         }
1059         if 0 <= strings.Index(string(line), rexpv[0]) {
1060             found += 1
1061             fmt.Printf("%s:%d: %s", path, li, line)
1062         }
1063     }
1064     //fmt.Printf("total %d lines %s\n", li, path)
1065     //if 0 < found { fmt.Printf("(found %d lines %s)\n", found, path); }
1066     return found
1067 }
1068
1069 // <a name="finder">Finder</a>
1070 // finding files with it name and contents
1071 // file names are ORed
1072 // show the content with %x fmt list
1073 // ls -R
1074 // tar command by adding output
1075 type fileSum struct {
1076     Err int64 // access error or so
1077     Size int64 // content size
1078     DupSize int64 // content size from hard links
1079     Blocks int64 // number of blocks (of 512 bytes)
1080     DupBlocks int64 // Blocks pointed from hard links
1081     HLinks int64 // hard links
1082     Words int64 // hard links
1083     Lines int64
1084     Files int64
1085     Dirs int64 // the num. of directories
1086     SymLink int64
1087     Flats int64 // the num. of flat files
1088     MaxDepth int64
1089     MaxNamlen int64 // max. name length
1090     nextRepo time.Time
1091 }
1092 func showFusage(dir string, fusage *fileSum){
1093     bsume := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1094     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1095
1096     fmt.Printf("%v: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
1097         dir,
1098         fusage.Files,
1099         fusage.Dirs,
1100         fusage.SymLink,
1101         fusage.HLinks,
1102         float64(fusage.Size)/1000000.0, bsume);
1103 }
1104 const (
1105     S_IFMT = 0170000
1106     S_IFCHR = 0020000
1107     S_IFDIR = 0040000
1108     S_IFREG = 0100000
1109     S_IFLNK = 0120000
1110     S_IFSOCK = 0140000
1111 )
1112 func cumFinfo(fsum *fileSum, path string, stater error, fstat syscall.Stat_t, argv[]string, verb bool)(*fileSum){
1113     now := time.Now()
1114     if time.Second <= now.Sub(fsum.nextRepo) {
1115         if !fsum.nextRepo.IsZero(){
1116             tstamp := now.Format(time.Stamp)

```

```

1117     showFusage(tstmp, fsum)
1118     }
1119     fsum.nextRepo = now.Add(time.Second)
1120 }
1121 if staterr != nil {
1122     fsum.Err += 1
1123     return fsum
1124 }
1125 fsum.Files += 1
1126 if l < fstat.Nlink {
1127     // must count only once...
1128     // at least ignore ones in the same directory
1129     //if finfo.Mode().IsRegular() {
1130     if (fstat.Mode & S_IFMT) == S_IFREG {
1131         fsum.HLinks += 1
1132         fsum.DupBlocks += int64(fstat.Blocks)
1133         //fmt.Printf("----Dup HardLink %v %s\n", fstat.Nlink, path)
1134     }
1135 }
1136 //fsum.Size += finfo.Size()
1137 fsum.Size += fstat.Size
1138 fsum.Blocks += int64(fstat.Blocks)
1139 //if verb { fmt.Printf("%8dBlk %s", fstat.Blocks/2, path) }
1140 if isin("-ls", argv){
1141     //if verb { fmt.Printf("%4d %8d ", fstat.Blksize, fstat.Blocks) }
1142     // fmt.Printf("%d\t", fstat.Blocks/2)
1143 }
1144 //if finfo.IsDir()
1145 if (fstat.Mode & S_IFMT) == S_IFDIR {
1146     fsum.Dirs += 1
1147 }
1148 //if (finfo.Mode() & os.ModeSymlink) != 0
1149 if (fstat.Mode & S_IFMT) == S_IFLNK {
1150     //if verb { fmt.Printf("symlink(%v,%s)\n", fstat.Mode, finfo.Name()) }
1151     //{ fmt.Printf("symlink(%o,%s)\n", fstat.Mode, finfo.Name()) }
1152     fsum.SymLink += 1
1153 }
1154 }
1155 }
1156 func (gsh*GshContext)xxFindEntv(depth int, total *fileSum, dir string, dstat syscall.Stat_t, ei int, entv []string, npatv []string, argv []string)(*fileSum){
1157     nols := isin("--grep", argv)
1158     // sort entv
1159     /*
1160     if isin("-t", argv){
1161         sort.Slice(filev, func(i, j int) bool {
1162             return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
1163         })
1164     }
1165     */
1166     /*
1167     if isin("-u", argv){
1168         sort.Slice(filev, func(i, j int) bool {
1169             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1170         })
1171     }
1172     if isin("-U", argv){
1173         sort.Slice(filev, func(i, j int) bool {
1174             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1175         })
1176     }
1177     */
1178     /*
1179     if isin("-S", argv){
1180         sort.Slice(filev, func(i, j int) bool {
1181             return filev[j].Size() < filev[i].Size()
1182         })
1183     }
1184     */
1185     for _, filename := range entv {
1186         for _, npat := range npatv {
1187             match := true
1188             if npat == "*" {
1189                 match = true
1190             }else{
1191                 match, _ = filepath.Match(npat, filename)
1192             }
1193             path := dir + DIRSEP + filename
1194             if !match {
1195                 continue
1196             }
1197             var fstat syscall.Stat_t
1198             staterr := syscall.Lstat(path, &fstat)
1199             if staterr != nil {
1200                 if !isin("-w", argv){fmt.Printf("ufind: %v\n", staterr) }
1201                 continue;
1202             }
1203             if isin("-du", argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
1204                 // should not show size of directory in "-du" mode ...
1205             }else
1206             if !nols && !isin("-s", argv) && (!isin("-du", argv) || isin("-a", argv)) {
1207                 if isin("-du", argv) {
1208                     fmt.Printf("%d\t", fstat.Blocks/2)
1209                 }
1210                 showFileInfo(path, argv)
1211             }
1212             if true { // && isin("-du", argv)
1213                 total = cumFinfo(total, path, staterr, fstat, argv, false)
1214             }
1215             /*
1216             if isin("-wc", argv) {
1217                 }
1218             */
1219             if gsh.lastCheckSum.SumType != 0 {
1220                 gsh.xChecksum(path, argv, &gsh.lastCheckSum);
1221             }
1222             x := isinX("--grep", argv); // -grep will be convenient like -ls
1223             if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
1224                 if !isRegFile(path){
1225                     found := gsh.xGrep(path, argv[x+1:])
1226                     if 0 < found {
1227                         foundv := gsh.CmdCurrent.FoundFile
1228                         if len(foundv) < 10 {
1229                             gsh.CmdCurrent.FoundFile =
1230                                 append(gsh.CmdCurrent.FoundFile, path)
1231                         }
1232                     }
1233                 }
1234             }
1235             if !isin("-r0", argv) { // -d 0 in du, -depth n in find
1236                 //total.Depth += 1
1237                 if (fstat.Mode & S_IFMT) == S_IFLNK {
1238                     continue
1239                 }
1240                 if dstat.Rdev != fstat.Rdev {

```

```

1241         fmt.Printf("--I-- don't follow different device %v(%v) %v(%v)\n",
1242             dir,dstat.Rdev,path,fstat.Rdev)
1243     }
1244     if (fstat.Mode & S_IFMT) == S_IFDIR {
1245         total = gsh.xxFind(depth+1,total,path,npatv,argv)
1246     }
1247 }
1248 }
1249 }
1250 return total
1251 }
1252 func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npatv[]string,argv[]string)(*fileSum){
1253     nols := isin("-grep",argv)
1254     dirfile,oerr := os.OpenFile(dir,os.O_RDONLY,0)
1255     if oerr == nil {
1256         //fmt.Printf("--I-- %v(%v)[%d]\n",dir,dirfile,dirfile.Fd())
1257         defer dirfile.Close()
1258     }else{
1259     }
1260
1261     prev := *total
1262     var dstat syscall.Stat_t
1263     staterr := syscall.Lstat(dir,&dstat) // should be flstat
1264
1265     if staterr != nil {
1266         if !isin("-w",argv){ fmt.Printf("ufind: %v\n",staterr) }
1267         return total
1268     }
1269     //filev,err := ioutil.ReadDir(dir)
1270     //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1271     /*
1272     if err != nil {
1273         if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1274         return total
1275     }
1276     */
1277     if depth == 0 {
1278         total = cumFinfo(total,dir,staterr,dstat,argv,true)
1279         if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1280             showFileInfo(dir,argv)
1281         }
1282     }
1283     // it it is not a directory, just scan it and finish
1284
1285     for ei := 0; ; ei++ {
1286         entv,rderr := dirfile.Readdirnames(8*1024)
1287         if len(entv) == 0 || rderr != nil {
1288             //if rderr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),rderr) }
1289             break
1290         }
1291         if 0 < ei {
1292             fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
1293         }
1294         total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
1295     }
1296     if isin("-du",argv) {
1297         // if in "du" mode
1298         fmt.Printf("%d\t%s\n",total.Blocks-prev.Blocks)/2,dir)
1299     }
1300     return total
1301 }
1302
1303 // {ufind|fu|ls} [Files] [-- Names] [-- Expressions]
1304 // Files is "." by default
1305 // Names is "*" by default
1306 // Expressions is "-print" by default for "ufind", or -du for "fu" command
1307 func (gsh*GshContext)xFind(argv[]string){
1308     if 0 < len(argv) && strBegins(argv[0],"?"){
1309         showFound(gsh,argv)
1310         return
1311     }
1312     if isin("-cksum",argv) || isin("-sum",argv) {
1313         gsh.lastCheckSum = CheckSum{}
1314         if isin("-sum",argv) && isin("-add",argv) {
1315             gsh.lastCheckSum.SumType |= SUM_SUM64
1316         }else
1317         if isin("-sum",argv) && isin("-size",argv) {
1318             gsh.lastCheckSum.SumType |= SUM_SIZE
1319         }else
1320         if isin("-sum",argv) && isin("-bsd",argv) {
1321             gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
1322         }else
1323         if isin("-sum",argv) && isin("-sysv",argv) {
1324             gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
1325         }else
1326         if isin("-sum",argv) {
1327             gsh.lastCheckSum.SumType |= SUM_SUM64
1328         }
1329         if isin("-unix",argv) {
1330             gsh.lastCheckSum.SumType |= SUM_UNIXFILE
1331             gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32UNIX)
1332         }
1333         if isin("-ieee",argv){
1334             gsh.lastCheckSum.SumType |= SUM_CRCIEEE
1335             gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32IEEE)
1336         }
1337         gsh.lastCheckSum.RusgAtStart = Getrusagev()
1338     }
1339     var total = fileSum{}
1340     npats := []string{}
1341     for _,v := range argv {
1342         if 0 < len(v) && v[0] != '-' {
1343             npats = append(npats,v)
1344         }
1345         if v == "/" { break }
1346         if v == "-" { break }
1347         if v == "-grep" { break }
1348         if v == "-ls" { break }
1349     }
1350     if len(npats) == 0 {
1351         npats = []string{"*"}
1352     }
1353     cwd := "."
1354     // if to be fullpath :: cwd, _ := os.Getwd()
1355     if len(npats) == 0 { npats = []string{"*"} }
1356     fusage := gsh.xxFind(0,&total,cwd,npats,argv)
1357     if gsh.lastCheckSum.SumType != 0 {
1358         var sumi uint64 = 0
1359         sum := &gsh.lastCheckSum
1360         if (sum.SumType & SUM_SIZE) != 0 {
1361             sumi = uint64(sum.Size)
1362         }
1363         if (sum.SumType & SUM_SUM64) != 0 {
1364             sumi = sum.Sum64
1365         }
1366     }

```

```

1365     }
1366     if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1367         s := uint32(sum.Sum16)
1368         r := (s & 0xFFFF) + ((s & 0xFFFFFFFF) >> 16)
1369         s = (r & 0xFFFF) + (r >> 16)
1370         sum.Crc32Val = uint32(s)
1371         sumi = uint64(s)
1372     }
1373     if (sum.SumType & SUM_SUM16_BSD) != 0 {
1374         sum.Crc32Val = uint32(sum.Sum16)
1375         sumi = uint64(sum.Sum16)
1376     }
1377     if (sum.SumType & SUM_UNIXFILE) != 0 {
1378         sum.Crc32Val = byteCRC32end(sum.Crc32Val, uint64(sum.Size))
1379         sumi = uint64(byteCRC32end(sum.Crc32Val, uint64(sum.Size)))
1380     }
1381     if l < sum.Files {
1382         fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
1383             sumi, sum.Size,
1384             abssize(sum.Size), sum.Files,
1385             abssize(sum.Size/sum.Files))
1386     }else{
1387         fmt.Printf("%v %v %v\n",
1388             sumi, sum.Size, npats[0])
1389     }
1390 }
1391 if !isin("-grep", argv) {
1392     showFusage("total", fusage)
1393 }
1394 if !isin("-s", argv){
1395     hits := len(gsh.CmdCurrent.FoundFile)
1396     if 0 < hits {
1397         fmt.Printf("--I-- %d files hits // can be refered with !%df\n",
1398             hits, len(gsh.CommandHistory))
1399     }
1400 }
1401 if gsh.lastCheckSum.SumType != 0 {
1402     if isin("-ru", argv) {
1403         sum := gsh.lastCheckSum
1404         sum.Done = time.Now()
1405         gsh.lastCheckSum.RusgAtEnd = Getrusagev()
1406         elps := sum.Done.Sub(sum.Start)
1407         fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\r\n",
1408             sum.Size, abssize(sum.Size), sum.Files, abssize(sum.Size/sum.Files))
1409         nanos := int64(elps)
1410         fmt.Printf("--cksum-time: %v/total, %v/file, %.1f files/s, %v\r\n",
1411             abtime(nanos),
1412             abtime(nanos/sum.Files),
1413             (float64(sum.Files)*1000000000.0)/float64(nanos),
1414             abbspeed(sum.Size, nanos))
1415         diff := RusageSubv(sum.RusgAtEnd, sum.RusgAtStart)
1416         fmt.Printf("--cksum-rusg: %v\n", sRusagef("", argv, diff))
1417     }
1418 }
1419 return
1420 }
1421 }
1422 func showFiles(files []string){
1423     sp := ""
1424     for i, file := range files {
1425         if 0 < i { sp = " " } else { sp = "" }
1426         fmt.Printf(sp+"%s", escapeWhiteSP(file))
1427     }
1428 }
1429 func showFound(gshCtx *GshContext, argv []string){
1430     for i, v := range gshCtx.CommandHistory {
1431         if 0 < len(v.FoundFile) {
1432             fmt.Printf(":%d (%d) ", i, len(v.FoundFile))
1433             if isin("-ls", argv){
1434                 fmt.Printf("\n")
1435                 for _, file := range v.FoundFile {
1436                     fmt.Printf(" ") //sub number?
1437                     showFileInfo(file, argv)
1438                 }
1439             }else{
1440                 showFiles(v.FoundFile)
1441                 fmt.Printf("\n")
1442             }
1443         }
1444     }
1445 }
1446 }
1447 func showMatchFile(filev []os.FileInfo, npat, dir string, argv []string)(string, bool){
1448     fname := ""
1449     found := false
1450     for _, v := range filev {
1451         match, _ := filepath.Match(npat, (v.Name()))
1452         if match {
1453             fname = v.Name()
1454             found = true
1455             //fmt.Printf("[%d] %s\n", i, v.Name())
1456             showIfExecutable(fname, dir, argv)
1457         }
1458     }
1459     return fname, found
1460 }
1461 func showIfExecutable(name, dir string, argv []string)(ffullpath string, ffound bool){
1462     var fullpath string
1463     if strBegins(name, DIRSEP){
1464         fullpath = name
1465     }else{
1466         fullpath = dir + DIRSEP + name
1467     }
1468     fi, err := os.Stat(fullpath)
1469     if err != nil {
1470         fullpath = dir + DIRSEP + name + ".go"
1471         fi, err = os.Stat(fullpath)
1472     }
1473     if err == nil {
1474         fm := fi.Mode()
1475         if fm.IsRegular() {
1476             // R_OK=4, W_OK=2, X_OK=1, F_OK=0
1477             if syscall.Access(fullpath, 5) == nil {
1478                 ffullpath = fullpath
1479                 ffound = true
1480                 if ! isin("-s", argv) {
1481                     showFileInfo(fullpath, argv)
1482                 }
1483             }
1484         }
1485     }
1486     return ffullpath, ffound
1487 }
1488 func which(list string, argv []string) (fullpathv []string, itis bool){

```

```

1489     if len(argv) <= 1 {
1490         fmt.Printf("Usage: which comand [-s] [-a] [-ls]\n")
1491         return []string(""), false
1492     }
1493     path := argv[1]
1494     if strBegins(path, "/") {
1495         // should check if executable?
1496         _, exOK := showIfExecutable(path, "/", argv)
1497         fmt.Printf("--D-- %v exOK=%v\n", path, exOK)
1498         return []string{path}, exOK
1499     }
1500     pathenv, efound := os.LookupEnv(list)
1501     if ! efound {
1502         fmt.Printf("--E-- which: no \"%s\" environment\n", list)
1503         return []string(""), false
1504     }
1505     showall := isin("-a", argv) || 0 <= strings.Index(path, "*")
1506     dirv := strings.Split(pathenv, PATHSEP)
1507     ffound := false
1508     ffullpath := path
1509     for _, dir := range dirv {
1510         if 0 <= strings.Index(path, "*") { // by wild-card
1511             list, _ := ioutil.ReadDir(dir)
1512             ffullpath, ffound = showMatchFile(list, path, dir, argv)
1513         } else {
1514             ffullpath, ffound = showIfExecutable(path, dir, argv)
1515         }
1516         //if ffound && !isin("-a", argv) {
1517         if ffound && !showall {
1518             break;
1519         }
1520     }
1521     return []string{ffullpath}, ffound
1522 }
1523
1524 func stripLeadingWSParg(argv []string) ([]string){
1525     for ; 0 < len(argv); {
1526         if len(argv[0]) == 0 {
1527             argv = argv[1:]
1528         } else {
1529             break
1530         }
1531     }
1532     return argv
1533 }
1534 func xEval(argv []string, nlend bool){
1535     argv = stripLeadingWSParg(argv)
1536     if len(argv) == 0 {
1537         fmt.Printf("eval [%%format] [Go-expression]\n")
1538         return
1539     }
1540     pfmt := "%v"
1541     if argv[0][0] == '%' {
1542         pfmt = argv[0]
1543         argv = argv[1:]
1544     }
1545     if len(argv) == 0 {
1546         return
1547     }
1548     gocode := strings.Join(argv, " ");
1549     //fmt.Printf("eval [%v] [%v]\n", pfmt, gocode)
1550     fset := token.NewFileSet()
1551     rval, _ := types.Eval(fset, nil, token.NoPos, gocode)
1552     fmt.Printf(pfmt, rval.Value)
1553     if nlend { fmt.Printf("\n") }
1554 }
1555
1556 func getval(name string) (found bool, val int) {
1557     /* should expand the name here */
1558     if name == "gsh.pid" {
1559         return true, os.Getpid()
1560     } else {
1561         if name == "gsh.ppid" {
1562             return true, os.Getppid()
1563         }
1564         return false, 0
1565     }
1566 }
1567 func echo(argv []string, nlend bool){
1568     for ai := 1; ai < len(argv); ai++ {
1569         if 1 < ai {
1570             fmt.Printf(" ");
1571         }
1572         arg := argv[ai]
1573         found, val := getval(arg)
1574         if found {
1575             fmt.Printf("%d", val)
1576         } else {
1577             fmt.Printf("%s", arg)
1578         }
1579     }
1580     if nlend {
1581         fmt.Printf("\n");
1582     }
1583 }
1584
1585 func resfile() string {
1586     return "gsh.tmp"
1587 }
1588 //var resF *File
1589 func resmap() {
1590     //_, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
1591     // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
1592     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
1593     if err != nil {
1594         fmt.Printf("refF could not open: %s\n", err)
1595     } else {
1596         fmt.Printf("refF opened\n")
1597     }
1598 }
1599
1600 // @@2020-0821
1601 func gshScanArg(str string, strip int)(argv []string){
1602     var si = 0
1603     var sb = 0
1604     var inBracket = 0
1605     var arg1 = make([]byte, LINESIZE)
1606     var ax = 0
1607     debug := false
1608
1609     for ; si < len(str); si++ {
1610         if str[si] != ' ' {
1611             break
1612         }
1612     }

```

```

1613     }
1614     sb = si
1615     for ; si < len(str); si++ {
1616         if sb <= si {
1617             if debug {
1618                 fmt.Printf("--Da- +%d %2d-%2d %s ... %s\n",
1619                     inBracket,sb,si,argv[0:ax],str[si:])
1620             }
1621         }
1622         ch := str[si]
1623         if ch == '{' {
1624             inBracket += 1
1625             if 0 < strip && inBracket <= strip {
1626                 //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
1627                 continue
1628             }
1629         }
1630         if 0 < inBracket {
1631             if ch == '}' {
1632                 inBracket -= 1
1633                 if 0 < strip && inBracket < strip {
1634                     //fmt.Printf("stripLEV %d < %d?\n",inBracket,strip)
1635                     continue
1636                 }
1637             }
1638             argv[ax] = ch
1639             ax += 1
1640             continue
1641         }
1642         if str[si] == ' ' {
1643             argv = append(argv,string(argv[0:ax]))
1644             if debug {
1645                 fmt.Printf("--Da- [%v][%-v] %s ... %s\n",
1646                     -1+len(argv),sb,si,string(argv[0:ax]),string(str[si:]))
1647             }
1648             sb = si+1
1649             ax = 0
1650             continue
1651         }
1652         argv[ax] = ch
1653         ax += 1
1654     }
1655     if sb < si {
1656         argv = append(argv,string(argv[0:ax]))
1657         if debug {
1658             fmt.Printf("--Da- [%v][%-v] %s ... %s\n",
1659                 -1+len(argv),sb,si,string(argv[0:ax]),string(str[si:]))
1660         }
1661     }
1662     if debug {
1663         fmt.Printf("--Da- %d [%s] => [%d]%\n",strip,str,len(argv),argv)
1664     }
1665     return argv
1666 }
1667
1668 // should get stderr (into tmpfile ?) and return
1669 func (gsh*GshContext)Open(name,mode string)(pin*os.File,pout*os.File,err bool){
1670     var pv = [int{-1,-1}]
1671     syscall.Pipe(pv)
1672
1673     xarg := gshScanArg(name,1)
1674     name = strings.Join(xarg, " ")
1675
1676     pin = os.NewFile(uintptr(pv[0]),"StdoutOf-"+name+"")
1677     pout = os.NewFile(uintptr(pv[1]),"StdinOf-"+name+"")
1678     fdix := 0
1679     dir := "?"
1680     if mode == "r" {
1681         dir = "<"
1682         fdix = 1 // read from the stdout of the process
1683     }else{
1684         dir = ">"
1685         fdix = 0 // write to the stdin of the process
1686     }
1687     gshPA := gsh.gshPA
1688     savfd := gshPA.Files[fdix]
1689
1690     var fd uintptr = 0
1691     if mode == "r" {
1692         fd = pout.Fd()
1693         gshPA.Files[fdix] = pout.Fd()
1694     }else{
1695         fd = pin.Fd()
1696         gshPA.Files[fdix] = pin.Fd()
1697     }
1698     // should do this by Goroutine?
1699     if false {
1700         fmt.Printf("---Ip- Opened fd[%v] %s %v\n",fd,dir,name)
1701         fmt.Printf("--RED1 [%d,%d,%d]->[%d,%d,%d]\n",
1702             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
1703             pin.Fd(),pout.Fd(),pout.Fd())
1704     }
1705     savi := os.Stdin
1706     savo := os.Stdout
1707     save := os.Stderr
1708     os.Stdin = pin
1709     os.Stdout = pout
1710     os.Stderr = pout
1711     gsh.Background = true
1712     gsh.gshelllh(name)
1713     gsh.Background = false
1714     os.Stdin = savi
1715     os.Stdout = savo
1716     os.Stderr = save
1717
1718     gshPA.Files[fdix] = savfd
1719     return pin,pout,false
1720 }
1721
1722 // <a name="ex-commands">External commands</a>
1723 func (gsh*GshContext)excommand(exec bool, argv []string) (notf bool,exit bool) {
1724     if gsh.CmdTrace { fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }
1725
1726     gshPA := gsh.gshPA
1727     fullpathv, itis := which("PATH",[]string{"which",argv[0],"-s"})
1728     if itis == false {
1729         return true,false
1730     }
1731     fullpath := fullpathv[0]
1732     argv = unescapeWhiteSPV(argv)
1733     if 0 < strings.Index(fullpath, ".go") {
1734         nargv := argv // []string{}
1735         gofullpathv, itis := which("PATH",[]string{"which","go","-s"})
1736         if itis == false {

```

```

1737         fmt.Printf("--F-- Go not found\n")
1738         return false,true
1739     }
1740     gofullpath := gofullpathv[0]
1741     nargsv = []string{ gofullpath, "run", fullpath }
1742     fmt.Printf("--I-- %s %s %s\n", gofullpath,
1743         nargsv[0],nargsv[1],nargsv[2])
1744     if exec {
1745         syscall.Exec(gofullpath,nargsv,os.Environ())
1746     }else{
1747         pid, _ := syscall.ForkExec(gofullpath,nargsv,&gshPA)
1748         if gsh.Background {
1749             fmt.Fprintf(stderr,"--Ip- in Background pid[%d]d(%v)\n",pid,len(nargsv),nargsv)
1750             gsh.BackgroundJobs = append(gsh.BackgroundJobs,pid)
1751         }else{
1752             rusage := syscall.Rusage {}
1753             syscall.Wait4(pid,nil,0,&rusage)
1754             gsh.LastRusage = rusage
1755             gsh.CmdCurrent.Rusagev[1] = rusage
1756         }
1757     }
1758 }else{
1759     if exec {
1760         syscall.Exec(fullpath,argv,os.Environ())
1761     }else{
1762         pid, _ := syscall.ForkExec(fullpath,argv,&gshPA)
1763         //fmt.Printf("[%d]\n",pid); // '&' to be background
1764         if gsh.Background {
1765             fmt.Fprintf(stderr,"--Ip- in Background pid[%d]d(%v)\n",pid,len(argv),argv)
1766             gsh.BackgroundJobs = append(gsh.BackgroundJobs,pid)
1767         }else{
1768             rusage := syscall.Rusage {}
1769             syscall.Wait4(pid,nil,0,&rusage);
1770             gsh.LastRusage = rusage
1771             gsh.CmdCurrent.Rusagev[1] = rusage
1772         }
1773     }
1774 }
1775     return false,false
1776 }
1777
1778 // <a name="builtin">Builtin Commands</a>
1779 func (gshCtx *GshContext) sleep(argv []string) {
1780     if len(argv) < 2 {
1781         fmt.Printf("Sleep 100ms, 100us, 100ns, ... \n")
1782         return
1783     }
1784     duration := argv[1];
1785     d, err := time.ParseDuration(duration)
1786     if err != nil {
1787         d, err = time.ParseDuration(duration+"s")
1788         if err != nil {
1789             fmt.Printf("duration ? %s (%s)\n",duration,err)
1790             return
1791         }
1792     }
1793     //fmt.Printf("Sleep %v\n",duration)
1794     time.Sleep(d)
1795     if 0 < len(argv[2:]) {
1796         gshCtx.gshellv(argv[2:])
1797     }
1798 }
1799 func (gshCtx *GshContext)repeat(argv []string) {
1800     if len(argv) < 2 {
1801         return
1802     }
1803     start0 := time.Now()
1804     for ri, _ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
1805         if 0 < len(argv[2:]) {
1806             //start := time.Now()
1807             gshCtx.gshellv(argv[2:])
1808             end := time.Now()
1809             elps := end.Sub(start0);
1810             if( 1000000000 < elps){
1811                 fmt.Printf("(repeat#%d %v)\n",ri,elps);
1812             }
1813         }
1814     }
1815 }
1816
1817 func (gshCtx *GshContext)gen(argv []string) {
1818     gshPA := gshCtx.gshPA
1819     if len(argv) < 2 {
1820         fmt.Printf("Usage: %s N\n",argv[0])
1821         return
1822     }
1823     // should br repeated by "repeat" command
1824     count, _ := strconv.Atoi(argv[1])
1825     fd := gshPA.Files[1] // Stdout
1826     file := os.NewFile(fd,"internalStdOut")
1827     fmt.Printf("--I-- Gen. Count=#d to [%d]\n",count,file.Fd())
1828     //buf := []byte{}
1829     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
1830     for gi := 0; gi < count; gi++ {
1831         file.WriteString(outdata)
1832     }
1833     //file.WriteString("\n")
1834     fmt.Printf("\n(%d B)\n",count*len(outdata));
1835     //file.Close()
1836 }
1837
1838 // <a name="rexec">Remote Execution</a> // 2020-0820
1839 func Elapsed(from time.Time)(string){
1840     elps := time.Now().Sub(from)
1841     if 1000000000 < elps {
1842         return fmt.Sprintf("[%5d.%02ds]",elps/1000000000,(elps%1000000000)/1000000)
1843     }else
1844     if 1000000 < elps {
1845         return fmt.Sprintf("[%3d.%03dms]",elps/1000000,(elps%1000000)/1000)
1846     }else{
1847         return fmt.Sprintf("[%3d.%03dus]",elps/1000,(elps%1000))
1848     }
1849 }
1850 func abftime(nanos int64)(string){
1851     if 1000000000 < nanos {
1852         return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%1000000000)/1000000)
1853     }else
1854     if 1000000 < nanos {
1855         return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
1856     }else{
1857         return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
1858     }
1859 }
1860 func abssize(size int64)(string){

```

```

1861     fsize := float64(size)
1862     if 1024*1024*1024 < size {
1863         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))
1864     }else
1865     if 1024*1024 < size {
1866         return fmt.Sprintf("%.3fMiB", fsize/(1024*1024))
1867     }else{
1868         return fmt.Sprintf("%.3fKiB", fsize/1024)
1869     }
1870 }
1871 func absize(size int64)(string){
1872     fsize := float64(size)
1873     if 1024*1024*1024 < size {
1874         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))
1875     }else
1876     if 1024*1024 < size {
1877         return fmt.Sprintf("%.3fMiB", fsize/(1024*1024))
1878     }else{
1879         return fmt.Sprintf("%.3fKiB", fsize/1024)
1880     }
1881 }
1882 func abbspeed(totalB int64, ns int64)(string){
1883     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
1884     if 1000 <= MBs {
1885         return fmt.Sprintf("%6.3fGB/s", MBs/1000)
1886     }
1887     if 1 <= MBs {
1888         return fmt.Sprintf("%6.3fMB/s", MBs)
1889     }else{
1890         return fmt.Sprintf("%6.3fKB/s", MBs*1000)
1891     }
1892 }
1893 func abspeed(totalB int64, ns time.Duration)(string){
1894     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
1895     if 1000 <= MBs {
1896         return fmt.Sprintf("%6.3fGBps", MBs/1000)
1897     }
1898     if 1 <= MBs {
1899         return fmt.Sprintf("%6.3fMBps", MBs)
1900     }else{
1901         return fmt.Sprintf("%6.3fKBps", MBs*1000)
1902     }
1903 }
1904 func fileRelay(what string, in*os.File, out*os.File, size int64, bsiz int)(wcount int64){
1905     Start := time.Now()
1906     buff := make([]byte, bsiz)
1907     var total int64 = 0
1908     var rem int64 = size
1909     nio := 0
1910     Prev := time.Now()
1911     var PrevSize int64 = 0
1912
1913     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) START\n",
1914         what, absize(total), size, nio)
1915
1916     for i:= 0; ; i++ {
1917         var len = bsiz
1918         if int(rem) < len {
1919             len = int(rem)
1920         }
1921         Now := time.Now()
1922         Elps := Now.Sub(Prev);
1923         if 1000000000 < Now.Sub(Prev) {
1924             fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %s\n",
1925                 what, absize(total), size, nio,
1926                 abspeed((total-PrevSize), Elps))
1927             Prev = Now;
1928             PrevSize = total
1929         }
1930         rlen := len
1931         if in != nil {
1932             // should watch the disconnection of out
1933             rcc, err := in.Read(buff[0:rlen])
1934             if err != nil {
1935                 fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<%v\n",
1936                     what, rcc, err, in.Name())
1937                 break
1938             }
1939             rlen = rcc
1940             if string(buff[0:10]) == "(SoftEOF " {
1941                 var ecc int64 = 0
1942                 fmt.Sscanf(string(buff), "(SoftEOF %v", &ecc)
1943                 fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))/%v\n",
1944                     what, ecc, total)
1945                 if ecc == total {
1946                     break
1947                 }
1948             }
1949         }
1950
1951         wlen := rlen
1952         if out != nil {
1953             wcc, err := out.Write(buff[0:rlen])
1954             if err != nil {
1955                 fmt.Printf(Elapsed(Start)+"--En-- X: %s write(%v,%v)>%v\n",
1956                     what, wcc, err, out.Name())
1957                 break
1958             }
1959             wlen = wcc
1960         }
1961         if wlen < rlen {
1962             fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
1963                 what, wlen, rlen)
1964             break;
1965         }
1966
1967         nio += 1
1968         total += int64(rlen)
1969         rem -= int64(rlen)
1970         if rem <= 0 {
1971             break
1972         }
1973     }
1974     Done := time.Now()
1975     Elps := float64(Done.Sub(Start))/1000000000 //Seconds
1976     TotalMB := float64(total)/1000000 //MB
1977     MBps := TotalMB / Elps
1978     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %v %3fMB/s\n",
1979         what, total, size, nio, absize(total), MBps)
1980     return total
1981 }
1982 func tcpPush(clnt *os.File){
1983     // shrink socket buffer and recover
1984     usleep(100);

```



```

1985 }
1986 func (gsh*GshContext)RexecServer(argv[]string){
1987     debug := true
1988     Start0 := time.Now()
1989     Start := Start0
1990     // if local == ":" { local = "0.0.0.0:9999" }
1991     local := "0.0.0.0:9999"
1992
1993     if 0 < len(argv) {
1994         if argv[0] == "-s" {
1995             debug = false
1996             argv = argv[1:]
1997         }
1998     }
1999     if 0 < len(argv) {
2000         argv = argv[1:]
2001     }
2002     port, err := net.ResolveTCPAddr("tcp",local);
2003     if err != nil {
2004         fmt.Printf("--En- S: Address error: %s (%s)\n",local,err)
2005         return
2006     }
2007     fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n",local);
2008     sconn, err := net.ListenTCP("tcp", port)
2009     if err != nil {
2010         fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n",local,err)
2011         return
2012     }
2013
2014     reqbuf := make([]byte,LINESIZE)
2015     res := ""
2016     for {
2017         fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
2018         aconn, err := sconn.AcceptTCP()
2019         Start = time.Now()
2020         if err != nil {
2021             fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
2022             return
2023         }
2024         clnt, _ := aconn.File()
2025         fd := clnt.Fd()
2026         ar := aconn.RemoteAddr()
2027         if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
2028             local,fd,ar) }
2029         res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
2030         fmt.Fprintf(clnt,"%s",res)
2031         if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
2032         count, err := clnt.Read(reqbuf)
2033         if err != nil {
2034             fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
2035                 count,err,string(reqbuf))
2036         }
2037         req := string(reqbuf[:count])
2038         if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
2039         reqv := strings.Split(string(req),"r")
2040         cmdv := gshScanArg(reqv[0],0)
2041         //cmdv := strings.Split(reqv[0]," ")
2042         switch cmdv[0] {
2043             case "HELO":
2044                 res = fmt.Sprintf("250 %v",req)
2045             case "GET":
2046                 // download {remotefile|-zN} [localfile]
2047                 var dsize int64 = 32*1024*1024
2048                 var bsize int = 64*1024
2049                 var fname string = ""
2050                 var in *os.File = nil
2051                 var pseudoEOF = false
2052                 if 1 < len(cmdv) {
2053                     fname = cmdv[1]
2054                     if strBegins(fname,"-z") {
2055                         fmt.Sscanf(fname[2:],"%d",&dsize)
2056                     }else
2057                     if strBegins(fname,"{") {
2058                         xin,xout,err := gsh.Popen(fname,"r")
2059                         if err {
2060                             }else{
2061                                 xout.Close()
2062                                 defer xin.Close()
2063                                 in = xin
2064                                 dsize = MaxStreamSize
2065                                 pseudoEOF = true
2066                             }
2067                     }else{
2068                         xin,err := os.Open(fname)
2069                         if err != nil {
2070                             fmt.Printf("--En- GET (%v)\n",err)
2071                         }else{
2072                             defer xin.Close()
2073                             in = xin
2074                             fi, _ := xin.Stat()
2075                             dsize = fi.Size()
2076                         }
2077                     }
2078                 }
2079                 //fmt.Printf(Elapsed(Start)+"--In- GET %v:%v\n",dsize,bsize)
2080                 res = fmt.Sprintf("200 %v\r\n",dsize)
2081                 fmt.Fprintf(clnt,"%v",res)
2082                 tcpPush(clnt); // should be separated as line in receiver
2083                 fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2084                 wcount := fileRelay("SendGET",in,clnt,dsize,bsize)
2085                 if pseudoEOF {
2086                     in.Close() // pipe from the command
2087                     // show end of stream data (its size) by OOB?
2088                     SoftEOF := fmt.Sprintf("((SoftEOF %v))",wcount)
2089                     fmt.Printf(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)
2090                 }
2091                 tcpPush(clnt); // to let SoftEOF data apper at the top of received data
2092                 fmt.Fprintf(clnt,"%v\r\n",SoftEOF)
2093                 tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
2094                 // with client generated random?
2095                 //fmt.Printf("--In- L: close %v (%v)\n",in.Fd(),in.Name())
2096             }
2097             res = fmt.Sprintf("200 GET done\r\n")
2098             case "PUT":
2099                 // upload {srcfile|-zN} [dstfile]
2100                 var dsize int64 = 32*1024*1024
2101                 var bsize int = 64*1024
2102                 var fname string = ""
2103                 var out *os.File = nil
2104                 if 1 < len(cmdv) { // localfile
2105                     fmt.Sscanf(cmdv[1],"%d",&dsize)
2106                 }
2107                 if 2 < len(cmdv) {
2108                     fname = cmdv[2]

```

```

2109         if fname == "-" {
2110             // nul dev
2111         }else
2112         if strBegins(fname,"{") {
2113             xin,xout,err := gsh.Popen(fname,"w")
2114             if err {
2115                 }else{
2116                     xin.Close()
2117                     defer xout.Close()
2118                     out = xout
2119                 }
2120             }else{
2121                 // should write to temporary file
2122                 // should suppress ^C on tty
2123                 xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2124                 //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n",fname,xout,err)
2125                 if err != nil {
2126                     fmt.Printf("--En- PUT (%v)\n",err)
2127                 }else{
2128                     out = xout
2129                 }
2130             }
2131             fmt.Printf(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
2132                 fname,local,err)
2133         }
2134         fmt.Printf(Elapsed(Start)+"--In- PUT %v (%v)\n",dsize,bsize)
2135         fmt.Printf(Elapsed(Start)+"--In- S: 200 %v OK\r\n",dsize)
2136         fmt.Fprintf(clnt,"200 %v OK\r\n",dsize)
2137         fileRelay("RecvPUT",clnt,out,dsize,bsize)
2138         res = fmt.Sprintf("200 PUT done\r\n")
2139         default:
2140             res = fmt.Sprintf("400 What? %v",req)
2141         }
2142         swcc,serr := clnt.Write([]byte(res))
2143         if serr != nil {
2144             fmt.Printf(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v",swcc,serr,res)
2145         }else{
2146             fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2147         }
2148         aconn.Close();
2149         clnt.Close();
2150     }
2151     sconn.Close();
2152 }
2153 func (gsh*GshContext)RexecClient(argv[]string)(int,string){
2154     debug := true
2155     Start := time.Now()
2156     if len(argv) == 1 {
2157         return -1,"EmptyARG"
2158     }
2159     argv = argv[1:]
2160     if argv[0] == "-serv" {
2161         gsh.RexecServer(argv[1:])
2162         return 0,"Server"
2163     }
2164     remote := "0.0.0.0:9999"
2165     if argv[0][0] == '@' {
2166         remote = argv[0][1:]
2167         argv = argv[1:]
2168     }
2169     if argv[0] == "-s" {
2170         debug = false
2171         argv = argv[1:]
2172     }
2173     dport, err := net.ResolveTCPAddr("tcp",remote);
2174     if err != nil {
2175         fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
2176         return -1,"AddressError"
2177     }
2178     fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
2179     serv, err := net.DialTCP("tcp",nil,dport)
2180     if err != nil {
2181         fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n",remote,err)
2182         return -1,"CannotConnect"
2183     }
2184     if debug {
2185         al := serv.LocalAddr()
2186         fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
2187     }
2188     req := ""
2189     res := make([]byte,LINESIZE)
2190     count,err := serv.Read(res)
2191     if err != nil {
2192         fmt.Printf("--En- S: (%3d,%v) %v",count,err,string(res))
2193     }
2194     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res)) }
2195     if argv[0] == "GET" {
2196         savPA := gsh.gshPA
2197         var bsize int = 64*1024
2198         req = fmt.Sprintf("%v\r\n",strings.Join(argv," "))
2199         fmt.Printf(Elapsed(Start)+"--In- C: %v",req)
2200         fmt.Fprintf(serv,req)
2201         count,err = serv.Read(res)
2202         if err != nil {
2203             }else{
2204                 var dsize int64 = 0
2205                 var out *os.File = nil
2206                 var out_tobeclosed *os.File = nil
2207                 var fname string = ""
2208                 var rcode int = 0
2209                 var pid int = -1
2210                 fmt.Sscanf(string(res),"%d %d",&rcode,&dsize)
2211                 fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
2212                 if 3 <= len(argv) {
2213                     fname = argv[2]
2214                     if strBegins(fname,"{") {
2215                         xin,xout,err := gsh.Popen(fname,"w")
2216                         if err {
2217                             }else{
2218                                 xin.Close()
2219                                 defer xout.Close()
2220                                 out = xout
2221                                 out_tobeclosed = xout
2222                                 pid = 0 // should be its pid
2223                             }
2224                         }else{
2225                             // should write to temporary file
2226                             // should suppress ^C on tty
2227                             xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2228                             if err != nil {
2229                                 fmt.Print("--En- %v\n",err)
2230                             }
2231                         }
2232                     }

```

```

2233         out = xout
2234         //fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
2235     }
2236 }
2237 in, _ := serv.File()
2238 fileRelay("RecvGET",in,out,dsize,bsize)
2239 if 0 <= pid {
2240     gsh.gshPA = savPA // recovery of Fd(), and more?
2241     fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
2242     out.tobclosed.Close()
2243     //syscall.Wait4(pid,nil,0,nil) //@@
2244 }
2245 }
2246 }else
2247 if argv[0] == "PUT" {
2248     remote, _ := serv.File()
2249     var local *os.File = nil
2250     var dsize int64 = 32*1024*1024
2251     var bsize int = 64*1024
2252     var ofile string = "-"
2253     //fmt.Printf("--I-- Rex %v\n",argv)
2254     if 1 < len(argv) {
2255         fname := argv[1]
2256         if strBegins(fname,"-z") {
2257             fmt.Sscanf(fname[2:],"%d",&dsize)
2258         }else
2259         if strBegins(fname,"{") {
2260             xin,xout,err := gsh.Popen(fname,"r")
2261             if err {
2262                 }else{
2263                     xout.Close()
2264                     defer xin.Close()
2265                     //in = xin
2266                     local = xin
2267                     fmt.Printf("--In- [%d] < Upload output of %v\n",
2268                         local.Fd(),fname)
2269                     ofile = "-from."+fname
2270                     dsize = MaxStreamSize
2271                 }
2272             }else{
2273                 xlocal,err := os.Open(fname)
2274                 if err != nil {
2275                     fmt.Printf("--En- (%s)\n",err)
2276                     local = nil
2277                 }else{
2278                     local = xlocal
2279                     fi, _ := local.Stat()
2280                     dsize = fi.Size()
2281                     defer local.Close()
2282                     //fmt.Printf("--I-- Rex in(%v / %v)\n",ofile,dsize)
2283                 }
2284                 ofile = fname
2285                 fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
2286                     fname,dsize,local,err)
2287             }
2288         }
2289         if 2 < len(argv) && argv[2] != "" {
2290             ofile = argv[2]
2291             //fmt.Printf("(%d)%v B.ofile=%v\n",len(argv),argv,ofile)
2292         }
2293         //fmt.Printf(Elapsed(Start)+"--I-- Rex out(%v)\n",ofile)
2294         fmt.Printf(Elapsed(Start)+"--In- PUT %v (/%v)\n",dsize,bsize)
2295         req = fmt.Sprintf("PUT %v %v \r\n",dsize,ofile)
2296         if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2297         fmt.Fprintf(serv,"%v",req)
2298         count,err = serv.Read(res)
2299         if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
2300         fileRelay("SendPUT",local,remote,dsize,bsize)
2301     }else{
2302         req = fmt.Sprintf("%v\r\n",strings.Join(argv," "))
2303         if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2304         fmt.Fprintf(serv,"%v",req)
2305         //fmt.Printf("--In- sending RexRequest(%v)\n",len(req))
2306     }
2307     //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
2308     count,err = serv.Read(res)
2309     ress := ""
2310     if count == 0 {
2311         ress = "(nil)\r\n"
2312     }else{
2313         ress = string(res[:count])
2314     }
2315     if err != nil {
2316         fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,ress)
2317     }else{
2318         fmt.Printf(Elapsed(Start)+"--In- S: %v",ress)
2319     }
2320     serv.Close()
2321     //conn.Close()
2322 }
2323 var stat string
2324 var rcode int
2325 fmt.Sscanf(ress,"%d %s",&rcode,&stat)
2326 //fmt.Printf("--D-- Client: %v (%v)",rcode,stat)
2327 return rcode,ress
2328 }
2329 }
2330 // <a name="remote-sh">Remote Shell</a>
2331 // gcp file [...] { [host]:[port]:[dir] | dir } // -p | -no-p
2332 func (gsh*GshContext)FileCopy(argv[]string){
2333     var host = ""
2334     var port = ""
2335     var upload = false
2336     var download = false
2337     var xargv = []string{"rex-gcp"}
2338     var srcv = []string{}
2339     var dstv = []string{}
2340     argv = argv[1:]
2341 }
2342 for _,v := range argv {
2343     /*
2344     if v[0] == '-' { // might be a pseudo file (generated date)
2345         continue
2346     }
2347     */
2348     obj := strings.Split(v," ")
2349     //fmt.Printf("%d %v %v\n",len(obj),v,obj)
2350     if 1 < len(obj) {
2351         host = obj[0]
2352         file := ""
2353         if 0 < len(host) {
2354             gsh.LastServer.host = host
2355         }else{
2356             host = gsh.LastServer.host

```

```

2357     port = gsh.LastServer.port
2358 }
2359 if 2 < len(obj) {
2360     port = obj[1]
2361     if 0 < len(port) {
2362         gsh.LastServer.port = port
2363     }else{
2364         port = gsh.LastServer.port
2365     }
2366     file = obj[2]
2367 }else{
2368     file = obj[1]
2369 }
2370 if len(srcv) == 0 {
2371     download = true
2372     srcv = append(srcv,file)
2373     continue
2374 }
2375 upload = true
2376 dstv = append(dstv,file)
2377 continue
2378 }
2379 /*
2380 idx := strings.Index(v,":")
2381 if 0 <= idx {
2382     remote = v[0:idx]
2383     if len(srcv) == 0 {
2384         download = true
2385         srcv = append(srcv,v[idx+1:])
2386         continue
2387     }
2388     upload = true
2389     dstv = append(dstv,v[idx+1:])
2390     continue
2391 }
2392 */
2393 if download {
2394     dstv = append(dstv,v)
2395 }else{
2396     srcv = append(srcv,v)
2397 }
2398 }
2399 hostport := "0" + host + ":" + port
2400 if upload {
2401     if host != "" { xargv = append(xargv,hostport) }
2402     xargv = append(xargv,"PUT")
2403     xargv = append(xargv,srcv[0]...)
2404     xargv = append(xargv,dstv[0]...)
2405     //fmt.Printf("--I-- FileCopy PUT gsh://%/< %v // %v\n",hostport,dstv,srcv,xargv)
2406     fmt.Printf("--I-- FileCopy PUT gsh://%/< %v\n",hostport,dstv,srcv)
2407     gsh.RexecClient(xargv)
2408 }else
2409 if download {
2410     if host != "" { xargv = append(xargv,hostport) }
2411     xargv = append(xargv,"GET")
2412     xargv = append(xargv,srcv[0]...)
2413     xargv = append(xargv,dstv[0]...)
2414     //fmt.Printf("--I-- FileCopy GET gsh://%/> %v // %v\n",hostport,srcv,dstv,xargv)
2415     fmt.Printf("--I-- FileCopy GET gsh://%/> %v\n",hostport,srcv,dstv)
2416     gsh.RexecClient(xargv)
2417 }else{
2418 }
2419 }
2420
2421 // target
2422 func (gsh*GshContext)Trelpath(rloc string)(string){
2423     cwd, _ := os.Getwd()
2424     os.Chdir(gsh.RWD)
2425     os.Chdir(rloc)
2426     twd, _ := os.Getwd()
2427     os.Chdir(cwd)
2428 }
2429 tpath := twd + "/" + rloc
2430 return tpath
2431 }
2432 // join to rmtote GShell - [user@]host[:port] or cd host[:port]:path
2433 func (gsh*GshContext)Rjoin(argv[]string){
2434     if len(argv) <= 1 {
2435         fmt.Printf("--I-- current server = %v\n",gsh.RSERV)
2436         return
2437     }
2438     serv := argv[1]
2439     servv := strings.Split(serv,":")
2440     if 1 <= len(servv) {
2441         if servv[0] == "lo" {
2442             servv[0] = "localhost"
2443         }
2444     }
2445     switch len(servv) {
2446     case 1:
2447         //if strings.Index(servv,"") < 0 {
2448             serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
2449         //}
2450     case 2: // host:port
2451         serv = strings.Join(servv,":")
2452     }
2453     xargv := []string{"rex-join","@"+serv,"HELO"}
2454     rcode,stat := gsh.RexecClient(xargv)
2455     if (rcode / 100) == 2 {
2456         fmt.Printf("--I-- OK Joined (%v) %v\n",rcode,stat)
2457         gsh.RSERV = serv
2458     }else{
2459         fmt.Printf("--I-- NG, could not joined (%v) %v\n",rcode,stat)
2460     }
2461 }
2462 func (gsh*GshContext)Rexec(argv[]string){
2463     if len(argv) <= 1 {
2464         fmt.Printf("--I-- rexec command [ | {file || {command} ]\n",gsh.RSERV)
2465         return
2466     }
2467 }
2468 /*
2469 nargv := gshScanArg(strings.Join(argv, " "),0)
2470 fmt.Printf("--D-- nargo=%d [%v]\n",len(nargv),nargv)
2471 if nargv[1][0] != '{' {
2472     nargv[1] = "{" + nargv[1] + "}"
2473     fmt.Printf("--D-- nargo=%d [%v]\n",len(nargv),nargv)
2474 }
2475 argv = nargv
2476 */
2477 nargv := []string{}
2478 nargv = append(nargv,"{"+strings.Join(argv[1:]," ")+"}")
2479 fmt.Printf("--D-- nargo=%d %v\n",len(nargv),nargv)
2480 argv = nargv

```

```

2481
2482     xargv := []string{"rex-exec", "@"+gsh.RSERV, "GET"}
2483     xargv = append(xargv, argv...)
2484     xargv = append(xargv, "/dev/tty")
2485     rcode, stat := gsh.RexecClient(xargv)
2486     if (rcode / 100) == 2 {
2487         fmt.Printf("--I-- OK Rexec (%v) %v\n", rcode, stat)
2488     } else {
2489         fmt.Printf("--I-- NG Rexec (%v) %v\n", rcode, stat)
2490     }
2491 }
2492 func (gsh*GshContext)Rchdir(argv []string){
2493     if len(argv) <= 1 {
2494         return
2495     }
2496     cwd, _ := os.Getwd()
2497     os.Chdir(gsh.RWD)
2498     os.Chdir(argv[1])
2499     twd, _ := os.Getwd()
2500     gsh.RWD = twd
2501     fmt.Printf("--I-- JWD=%v\n", twd)
2502     os.Chdir(cwd)
2503 }
2504 func (gsh*GshContext)Rpwd(argv []string){
2505     fmt.Printf("%v\n", gsh.RWD)
2506 }
2507 func (gsh*GshContext)Rls(argv []string){
2508     cwd, _ := os.Getwd()
2509     os.Chdir(gsh.RWD)
2510     argv[0] = "-ls"
2511     gsh.xFind(argv)
2512     os.Chdir(cwd)
2513 }
2514 func (gsh*GshContext)Rput(argv []string){
2515     var local string = ""
2516     var remote string = ""
2517     if 1 < len(argv) {
2518         local = argv[1]
2519         remote = local // base name
2520     }
2521     if 2 < len(argv) {
2522         remote = argv[2]
2523     }
2524     fmt.Printf("--I-- jput from=%v to=%v\n", local, gsh.Trelpath(remote))
2525 }
2526 func (gsh*GshContext)Rget(argv []string){
2527     var remote string = ""
2528     var local string = ""
2529     if 1 < len(argv) {
2530         remote = argv[1]
2531         local = remote // base name
2532     }
2533     if 2 < len(argv) {
2534         local = argv[2]
2535     }
2536     fmt.Printf("--I-- jget from=%v to=%v\n", gsh.Trelpath(remote), local)
2537 }
2538
2539 // <a name="network">network</a>
2540 // -s, -si, -so // bi-directional, source, sync (maybe socket)
2541 func (gshCtx*GshContext)connect(inTCP bool, argv []string) {
2542     gshPA := gshCtx.gshPA
2543     if len(argv) < 2 {
2544         fmt.Printf("Usage: -s [host]:[port[.udp]]\n")
2545         return
2546     }
2547     remote := argv[1]
2548     if remote == "" { remote = "0.0.0.0:9999" }
2549
2550     if inTCP { // TCP
2551         dport, err := net.ResolveTCPAddr("tcp", remote);
2552         if err != nil {
2553             fmt.Printf("Address error: %s (%s)\n", remote, err)
2554             return
2555         }
2556         conn, err := net.DialTCP("tcp", nil, dport)
2557         if err != nil {
2558             fmt.Printf("Connection error: %s (%s)\n", remote, err)
2559             return
2560         }
2561         file, _ := conn.File();
2562         fd := file.Fd()
2563         fmt.Printf("Socket: connected to %s, socket[%d]\n", remote, fd)
2564
2565         savfd := gshPA.Files[1]
2566         gshPA.Files[1] = fd;
2567         gshCtx.gshellv(argv[2:])
2568         gshPA.Files[1] = savfd
2569         file.Close()
2570         conn.Close()
2571     } else {
2572         //dport, err := net.ResolveUDPAddr("udp4", remote);
2573         dport, err := net.ResolveUDPAddr("udp", remote);
2574         if err != nil {
2575             fmt.Printf("Address error: %s (%s)\n", remote, err)
2576             return
2577         }
2578         //conn, err := net.DialUDP("udp4", nil, dport)
2579         conn, err := net.DialUDP("udp", nil, dport)
2580         if err != nil {
2581             fmt.Printf("Connection error: %s (%s)\n", remote, err)
2582             return
2583         }
2584         file, _ := conn.File();
2585         fd := file.Fd()
2586
2587         ar := conn.RemoteAddr()
2588         //al := conn.LocalAddr()
2589         fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
2590             remote, ar.String(), fd)
2591
2592         savfd := gshPA.Files[1]
2593         gshPA.Files[1] = fd;
2594         gshCtx.gshellv(argv[2:])
2595         gshPA.Files[1] = savfd
2596         file.Close()
2597         conn.Close()
2598     }
2599 }
2600 func (gshCtx*GshContext)saccept(inTCP bool, argv []string) {
2601     gshPA := gshCtx.gshPA
2602     if len(argv) < 2 {
2603         fmt.Printf("Usage: -ac [host]:[port[.udp]]\n")
2604         return

```

```

2605 }
2606 local := argv[1]
2607 if local == "" { local = "0.0.0.0:9999" }
2608 if inTCP { // TCP
2609     port, err := net.ResolveTCPAddr("tcp", local);
2610     if err != nil {
2611         fmt.Printf("Address error: %s (%s)\n", local, err)
2612         return
2613     }
2614     //fmt.Printf("Listen at %s...\n", local);
2615     sconn, err := net.ListenTCP("tcp", port)
2616     if err != nil {
2617         fmt.Printf("Listen error: %s (%s)\n", local, err)
2618         return
2619     }
2620     //fmt.Printf("Accepting at %s...\n", local);
2621     aconn, err := sconn.AcceptTCP()
2622     if err != nil {
2623         fmt.Printf("Accept error: %s (%s)\n", local, err)
2624         return
2625     }
2626     file, _ := aconn.File()
2627     fd := file.Fd()
2628     fmt.Printf("Accepted TCP at %s [%d]\n", local, fd)
2629
2630     savfd := gshPA.Files[0]
2631     gshPA.Files[0] = fd;
2632     gshCtx.gshellv(argv[2:])
2633     gshPA.Files[0] = savfd
2634
2635     sconn.Close();
2636     aconn.Close();
2637     file.Close();
2638 }else{
2639     //port, err := net.ResolveUDPAddr("udp4", local);
2640     port, err := net.ResolveUDPAddr("udp", local);
2641     if err != nil {
2642         fmt.Printf("Address error: %s (%s)\n", local, err)
2643         return
2644     }
2645     fmt.Printf("Listen UDP at %s...\n", local);
2646     //uconn, err := net.ListenUDP("udp4", port)
2647     uconn, err := net.ListenUDP("udp", port)
2648     if err != nil {
2649         fmt.Printf("Listen error: %s (%s)\n", local, err)
2650         return
2651     }
2652     file, _ := uconn.File()
2653     fd := file.Fd()
2654     ar := uconn.RemoteAddr()
2655     remote := ""
2656     if ar != nil { remote = ar.String() }
2657     if remote == "" { remote = "?" }
2658
2659     // not yet received
2660     //fmt.Printf("Accepted at %s [%d] <- %s\n", local, fd, "")
2661
2662     savfd := gshPA.Files[0]
2663     gshPA.Files[0] = fd;
2664     savenv := gshPA.Env
2665     gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
2666     gshCtx.gshellv(argv[2:])
2667     gshPA.Env = savenv
2668     gshPA.Files[0] = savfd
2669
2670     uconn.Close();
2671     file.Close();
2672 }
2673 }
2674
2675 // empty line command
2676 func (gshCtx *GshContext)xPvd(argv []string){
2677     // execute context command, pwd + date
2678     // context notation, representation scheme, to be resumed at re-login
2679     cwd, _ := os.Getwd()
2680     switch {
2681     case isin("-a", argv):
2682         gshCtx.ShowChdirHistory(argv)
2683     case isin("-ls", argv):
2684         showFileInfo(cwd, argv)
2685     default:
2686         fmt.Printf("%s\n", cwd)
2687     case isin("-v", argv): // obsolete empty command
2688         t := time.Now()
2689         date := t.Format(time.UnixDate)
2690         exe, _ := os.Executable()
2691         host, _ := os.Hostname()
2692         fmt.Printf("{PWD=\"%s\" ,cwd)
2693         fmt.Printf(" HOST=\"%s\" ,host)
2694         fmt.Printf(" DATE=\"%s\" ,date)
2695         fmt.Printf(" TIME=\"%s\" ,t.String())
2696         fmt.Printf(" PID=\"%d\" ,os.Getpid())
2697         fmt.Printf(" EXE=\"%s\" ,exe)
2698         fmt.Printf(")\n")
2699     }
2700 }
2701
2702 // <a name="history">History</a>
2703 // these should be browsed and edited by HTTP browser
2704 // show the time of command with -t and direcotry with -ls
2705 // openfile-history, sort by -a -m -c
2706 // sort by elapsed time by -t -s
2707 // search by "more" like interface
2708 // edit history
2709 // sort history, and wc or uniq
2710 // CPU and other resource consumptions
2711 // limit showing range (by time or so)
2712 // export / import history
2713 func (gshCtx *GshContext)xHistory(argv []string){
2714     atWorkDirX := -1
2715     if 1 < len(argv) && strBegins(argv[1], "8") {
2716         atWorkDirX, _ = strconv.Atoi(argv[1][1:])
2717     }
2718     //fmt.Printf("--D-- showHistory(%w)\n", argv)
2719     for i, v := range gshCtx.CommandHistory {
2720         // exclude commands not to be listed by default
2721         // internal commands may be suppressed by default
2722         if v.CmdLine == "" && !isin("-a", argv) {
2723             continue;
2724         }
2725         if 0 <= atWorkDirX {
2726             if v.WorkDirX != atWorkDirX {
2727                 continue
2728             }

```

```

2729     }
2730     if !isin("-n",argv){ // like "fc"
2731         fmt.Printf("!%2d ",i)
2732     }
2733     if isin("-v",argv){
2734         fmt.Println(v) // should be with it date
2735     }else{
2736         if isin("-l",argv) || isin("-l0",argv) {
2737             elps := v.EndAt.Sub(v.StartAt);
2738             start := v.StartAt.Format(time.Stamp)
2739             fmt.Printf("@%d ",v.WorkDirX)
2740             fmt.Printf("[%v] %11v/t ",start,elps)
2741         }
2742         if isin("-l",argv) && !isin("-l0",argv){
2743             fmt.Printf("%v",Rusagef("%t %u\t// %s",argv,v.Rusagev))
2744         }
2745         if isin("-at",argv) { // isin("-ls",argv){
2746             dhi := v.WorkDirX // workdir history index
2747             fmt.Printf("@%d %s\t",dhi,v.WorkDir)
2748             // show the FileInfo of the output command??
2749         }
2750         fmt.Printf("%s",v.CmdLine)
2751         fmt.Printf("\n")
2752     }
2753 }
2754 }
2755 // In - history index
2756 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
2757     if gline[0] == '!' {
2758         hix, err := strconv.Atoi(gline[1:])
2759         if err != nil {
2760             fmt.Printf("--E-- (%s : range)\n",hix)
2761             return "", false, true
2762         }
2763         if hix < 0 || len(gshCtx.CommandHistory) <= hix {
2764             fmt.Printf("--E-- (%d : out of range)\n",hix)
2765             return "", false, true
2766         }
2767         return gshCtx.CommandHistory[hix].CmdLine, false, false
2768     }
2769     // search
2770     //for i, v := range gshCtx.CommandHistory {
2771     //}
2772     return gline, false, false
2773 }
2774 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
2775     if 0 <= hix && hix < len(gsh.CommandHistory) {
2776         return gsh.CommandHistory[hix].CmdLine,true
2777     }
2778     return "",false
2779 }
2780 // temporary adding to PATH environment
2781 // cd name -lib for LD_LIBRARY_PATH
2782 // chdir with directory history (date + full-path)
2783 // -s for sort option (by visit date or so)
2784 func (gsh*GshContext)ShowChdirHistory(i int,v GChdirHistory, argv []string){
2785     fmt.Printf("!%2d ",v.CmdIndex) // the first command at this WorkDir
2786     fmt.Printf("@%d ",i)
2787     fmt.Printf("[%v] ",v.MovedAt.Format(time.Stamp))
2788     showFileInfo(v.Dir,argv)
2789 }
2790 func (gsh*GshContext)ShowChdirHistory(argv []string){
2791     for i, v := range gsh.ChdirHistory {
2792         gsh.ShowChdirHistory1(i,v,argv)
2793     }
2794 }
2795 }
2796 func skipOpts(argv[]string)(int){
2797     for i,v := range argv {
2798         if strBegins(v,"-") {
2799             }else{
2800                 return i
2801             }
2802     }
2803     return -1
2804 }
2805 func (gshCtx*GshContext)xChdir(argv []string){
2806     cdhist := gshCtx.ChdirHistory
2807     if isin("?",argv) || isin("-t",argv) || isin("-a",argv) {
2808         gshCtx.ShowChdirHistory(argv)
2809         return
2810     }
2811     pwd, _ := os.Getwd()
2812     dir := ""
2813     if len(argv) <= 1 {
2814         dir = toFullPath("--")
2815     }else{
2816         i := skipOpts(argv[1:])
2817         if i < 0 {
2818             dir = toFullPath("--")
2819         }else{
2820             dir = argv[1+i]
2821         }
2822     }
2823     if strBegins(dir,"@") {
2824         if dir == "@0" { // obsolete
2825             dir = gshCtx.StartDir
2826         }else
2827         if dir == "@!" {
2828             index := len(cdhist) - 1
2829             if 0 < index { index -= 1 }
2830             dir = cdhist[index].Dir
2831         }else{
2832             index, err := strconv.Atoi(dir[1:])
2833             if err != nil {
2834                 fmt.Printf("--E-- xChdir(%v)\n",err)
2835                 dir = "?"
2836             }else
2837             if len(gshCtx.ChdirHistory) <= index {
2838                 fmt.Printf("--E-- xChdir(history range error)\n")
2839                 dir = "?"
2840             }else{
2841                 dir = cdhist[index].Dir
2842             }
2843         }
2844     }
2845     if dir != "?" {
2846         err := os.Chdir(dir)
2847         if err != nil {
2848             fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
2849         }else{
2850             cwd, _ := os.Getwd()
2851             if cwd != pwd {
2852                 hist1 := GChdirHistory { }

```

```

2853     hist1.Dir = cwd
2854     hist1.MovedAt = time.Now()
2855     hist1.CmdIndex = len(gshCtx.CommandHistory)+1
2856     gshCtx.ChdirHistory = append(cdnhist,hist1)
2857     if !isin("-s",argv){
2858         //cwd, _ := os.Getwd()
2859         //fmt.Printf("%s\n",cwd)
2860         ix := len(gshCtx.ChdirHistory)-1
2861         gshCtx.ShowChdirHistory1(ix,hist1,argv)
2862     }
2863 }
2864 }
2865 }
2866 if isin("-ls",argv){
2867     cwd, _ := os.Getwd()
2868     showFileInfo(cwd,argv);
2869 }
2870 }
2871 func TimeValSub(tv1 *syscall.Timeval, tv2 *syscall.Timeval){
2872     *tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
2873 }
2874 func RusageSubv(ru1, ru2 [2]syscall.Rusage) ([2]syscall.Rusage){
2875     TimeValSub(&ru1[0].Utime,&ru2[0].Utime)
2876     TimeValSub(&ru1[0].Stime,&ru2[0].Stime)
2877     TimeValSub(&ru1[1].Utime,&ru2[1].Utime)
2878     TimeValSub(&ru1[1].Stime,&ru2[1].Stime)
2879     return ru1
2880 }
2881 func TimeValAdd(tv1 syscall.Timeval, tv2 syscall.Timeval)(syscall.Timeval){
2882     tvs := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
2883     return tvs
2884 }
2885 /*
2886 func RusageAddv(ru1, ru2 [2]syscall.Rusage) ([2]syscall.Rusage){
2887     TimeValAdd(ru1[0].Utime,ru2[0].Utime)
2888     TimeValAdd(ru1[0].Stime,ru2[0].Stime)
2889     TimeValAdd(ru1[1].Utime,ru2[1].Utime)
2890     TimeValAdd(ru1[1].Stime,ru2[1].Stime)
2891     return ru1
2892 }
2893 */
2894 // <a name="rusage">Resource Usage</a>
2895 func sRusage(fmtspec string, argv []string, ru [2]syscall.Rusage)(string){
2896     // ru[0] self , ru[1] children
2897     ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
2898     st := TimeValAdd(ru[0].Stime,ru[1].Stime)
2899     uu := (ut.Sec*1000000 + int64(ut.Usec)) * 1000
2900     su := (st.Sec*1000000 + int64(st.Usec)) * 1000
2901     tu := uu + su
2902     ret := fmt.Sprintf("%v/sum",abftime(tu))
2903     ret += fmt.Sprintf(", %v/usr",abftime(uu))
2904     ret += fmt.Sprintf(", %v/sys",abftime(su))
2905     return ret
2906 }
2907 }
2908 func Rusagef(fmtspec string, argv []string, ru [2]syscall.Rusage)(string){
2909     ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
2910     st := TimeValAdd(ru[0].Stime,ru[1].Stime)
2911     fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
2912     fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
2913     return ""
2914 }
2915 func Getrusagev()([2]syscall.Rusage){
2916     var ruv = [2]syscall.Rusage{}
2917     syscall.Getrusage(syscall.RUSAGE_SELF,&ruv[0])
2918     syscall.Getrusage(syscall.RUSAGE_CHILDREN,&ruv[1])
2919     return ruv
2920 }
2921 func showRusage(what string,argv []string, ru *syscall.Rusage){
2922     fmt.Printf("%s: ",what);
2923     fmt.Printf(" Utr=%d.%06ds",ru.Utime.Sec,ru.Utime.Usec)
2924     fmt.Printf(" Sys=%d.%06ds",ru.Stime.Sec,ru.Stime.Usec)
2925     fmt.Printf(" Rss=%vB",ru.Maxrss)
2926     if isin("-l",argv) {
2927         fmt.Printf(" MinFlt=%v",ru.Minflt)
2928         fmt.Printf(" MajFlt=%v",ru.Majflt)
2929         fmt.Printf(" IxRSS=%vB",ru.Ixrss)
2930         fmt.Printf(" IdRSS=%vB",ru.Idrss)
2931         fmt.Printf(" Nswap=%vB",ru.Nswap)
2932         fmt.Printf(" Read=%v",ru.Inblock)
2933         fmt.Printf(" Write=%v",ru.Oublock)
2934     }
2935     fmt.Printf(" Snd=%v",ru.Msgsnd)
2936     fmt.Printf(" Rcv=%v",ru.Msgrcv)
2937     //if isin("-l",argv) {
2938         fmt.Printf(" Sig=%v",ru.Nsignals)
2939     //}
2940     fmt.Printf("\n");
2941 }
2942 }
2943 func (gshCtx *GshContext)xTime(argv []string)(bool){
2944     if 2 <= len(argv){
2945         gshCtx.LastRusage = syscall.Rusage{}
2946         rusagev1 := Getrusagev()
2947         fin := gshCtx.gshellv(argv[1:])
2948         rusagev2 := Getrusagev()
2949         showRusage(argv[1],argv,&gshCtx.LastRusage)
2950         rusagev := RusageSubv(rusagev2,rusagev1)
2951         showRusage("self",argv,&rusagev[0])
2952         showRusage("chld",argv,&rusagev[1])
2953         return fin
2954     }else{
2955         rusage:= syscall.Rusage {}
2956         syscall.Getrusage(syscall.RUSAGE_SELF,&rusage)
2957         showRusage("self",argv, &rusage)
2958         syscall.Getrusage(syscall.RUSAGE_CHILDREN,&rusage)
2959         showRusage("chld",argv, &rusage)
2960         return false
2961     }
2962 }
2963 }
2964 func (gshCtx *GshContext)xJobs(argv []string){
2965     fmt.Printf("%d Jobs\n",len(gshCtx.BackGroundJobs))
2966     for ji, pid := range gshCtx.BackGroundJobs {
2967         //wstat := syscall.WaitStatus {0}
2968         rusage := syscall.Rusage {}
2969         //wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&rusage);
2970         wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
2971         if err != nil {
2972             fmt.Printf("--E-- %%%d [%d] (%v)\n",ji,pid,err)
2973         }else{
2974             fmt.Printf("%%d[%d](%d)\n",ji,pid,wpid)
2975             showRusage("chld",argv,&rusage)
2976         }
2977     }
2978 }

```



```

2977 func (gsh*GshContext)inBackground(argv[]string)(bool){
2978     if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n",argv) }
2979     gsh.BackGround = true // set background option
2980     xfin := false
2981     xfin = gsh.gshellv(argv)
2982     gsh.BackGround = false
2983     return xfin
2984 }
2985 // -o file without command means just opening it and refer by #N
2986 // should be listed by "files" command
2987 func (gshCtx*GshContext)xOpen(argv[]string){
2988     var pv = []int{-1,-1}
2989     err := syscall.Pipe(pv)
2990     fmt.Printf("--I-- pipe()=[#%d,#%d](%v)\n",pv[0],pv[1],err)
2991 }
2992 func (gshCtx*GshContext)fromPipe(argv[]string){
2993 }
2994 func (gshCtx*GshContext)xClose(argv[]string){
2995 }
2996
2997 // <a name="redirect">redirect</a>
2998 func (gshCtx*GshContext)redirect(argv[]string)(bool){
2999     if len(argv) < 2 {
3000         return false
3001     }
3002
3003     cmd := argv[0]
3004     fname := argv[1]
3005     var file *os.File = nil
3006
3007     fdix := 0
3008     mode := os.O_RDONLY
3009
3010     switch {
3011     case cmd == "-i" || cmd == "<":
3012         fdix = 0
3013         mode = os.O_RDONLY
3014     case cmd == "-o" || cmd == ">":
3015         fdix = 1
3016         mode = os.O_RDWR | os.O_CREATE
3017     case cmd == "-a" || cmd == ">>":
3018         fdix = 1
3019         mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3020     }
3021     if fname[0] == '#' {
3022         fd, err := strconv.Atoi(fname[1:])
3023         if err != nil {
3024             fmt.Printf("--E-- (%v)\n",err)
3025             return false
3026         }
3027         file = os.NewFile(uintptr(fd),"MaybePipe")
3028     }else{
3029         xfile, err := os.OpenFile(argv[1], mode, 0600)
3030         if err != nil {
3031             fmt.Printf("--E-- (%s)\n",err)
3032             return false
3033         }
3034         file = xfile
3035     }
3036     gshPA := gshCtx.gshPA
3037     savfd := gshPA.Files[fdix]
3038     gshPA.Files[fdix] = file.Fd()
3039     fmt.Printf("--I-- Opened [%d] %s\n",file.Fd(),argv[1])
3040     gshCtx.gshellv(argv[2:])
3041     gshPA.Files[fdix] = savfd
3042
3043     return false
3044 }
3045
3046 //fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
3047 func httpHandler(res http.ResponseWriter, req *http.Request){
3048     path := req.URL.Path
3049     fmt.Printf("--I-- Got HTTP Request(%s)\n",path)
3050     {
3051         gshCtxBuf, _ := setupGshContext()
3052         gshCtx := &gshCtxBuf
3053         fmt.Printf("--I-- %s\n",path[1:])
3054         gshCtx.tgshelll(path[1:])
3055     }
3056     fmt.Fprintf(res, "Hello(^_^)//\n%s\n",path)
3057 }
3058 func (gshCtx *GshContext) httpServer(argv []string){
3059     http.HandleFunc("/", httpHandler)
3060     accport := "localhost:9999"
3061     fmt.Printf("--I-- HTTP Server Start at [%s]\n",accport)
3062     http.ListenAndServe(accport,nil)
3063 }
3064 func (gshCtx *GshContext)xGo(argv[]string){
3065     go gshCtx.gshellv(argv[1:]);
3066 }
3067 func (gshCtx *GshContext) xPs(argv[]string)(){
3068 }
3069
3070 // <a name="plugin">Plugin</a>
3071 // plugin [-ls [names]] to list plugins
3072 // Reference: <a href="https://golang.org/src/plugin/">plugin</a> source code
3073 func (gshCtx *GshContext) whichPlugin(name string,argv[]string)(pi *PluginInfo){
3074     pi = nil
3075     for _,p := range gshCtx.PluginFuncs {
3076         if p.Name == name && pi == nil {
3077             pi = &p
3078         }
3079         if !isin("-s",argv){
3080             //fmt.Printf("%v %v ",i,p)
3081             if !isin("-ls",argv){
3082                 showFileInfo(p.Path,argv)
3083             }else{
3084                 fmt.Printf("%s\n",p.Name)
3085             }
3086         }
3087     }
3088     return pi
3089 }
3090 func (gshCtx *GshContext) xPlugin(argv[]string) (error) {
3091     if len(argv) == 0 || argv[0] == "-ls" {
3092         gshCtx.whichPlugin("",argv)
3093         return nil
3094     }
3095     name := argv[0]
3096     Pin := gshCtx.whichPlugin(name,[]string{"-s"})
3097     if Pin != nil {
3098         os.Args = argv // should be recovered?
3099         Pin.Addr.(func())()
3100         return nil

```

```

3101 }
3102 sofile := toFullpath(argv[0] + ".so") // or find it by which($PATH)
3103
3104 p, err := plugin.Open(sofile)
3105 if err != nil {
3106     fmt.Printf("--E-- plugin.Open(%s)(%v)\n", sofile, err)
3107     return err
3108 }
3109 fname := "Main"
3110 f, err := p.Lookup(fname)
3111 if( err != nil ){
3112     fmt.Printf("--E-- plugin.Lookup(%s)(%v)\n", fname, err)
3113     return err
3114 }
3115 pin := PluginInfo {p,f,name,sofile}
3116 gshCtx.PluginFuncs = append(gshCtx.PluginFuncs, pin)
3117 fmt.Printf("--I-- added (%d)\n", len(gshCtx.PluginFuncs))
3118
3119 //fmt.Printf("--I-- first call(%s:%s)%v\n", sofile, fname, argv)
3120 os.Args = argv
3121 f.(func())()
3122 return err
3123 }
3124 func (gshCtx*GshContext)Args(argv []string){
3125     for i,v := range os.Args {
3126         fmt.Printf("[%v] %v\n", i,v)
3127     }
3128 }
3129 func (gshCtx *GshContext) showVersion(argv []string){
3130     if isin("-l", argv) {
3131         fmt.Printf("%v/%v (%v)", NAME, VERSION, DATE);
3132     }else{
3133         fmt.Printf("%v", VERSION);
3134     }
3135     if isin("-a", argv) {
3136         fmt.Printf(" %s", AUTHOR)
3137     }
3138     if !isin("-n", argv) {
3139         fmt.Printf("\n")
3140     }
3141 }
3142
3143 // <a name="scanf">Scanf</a> // string decomposer
3144 // scanf [format] [input]
3145 func scanv(sstr string)(strv []string){
3146     strv = strings.Split(sstr, " ")
3147     return strv
3148 }
3149 func scanUntil(src, end string)(rstr string, leng int){
3150     idx := strings.Index(src, end)
3151     if 0 <= idx {
3152         rstr = src[0:idx]
3153         return rstr, idx+leng(end)
3154     }
3155     return src, 0
3156 }
3157
3158 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
3159 func (gsh*GshContext)printVal(fmts string, vstr string, optv []string){
3160     //vint, err := strconv.Atoi(vstr)
3161     var ival int64 = 0
3162     n := 0
3163     err := error(nil)
3164     if strBegins(vstr, "-") {
3165         vx, _ := strconv.Atoi(vstr[1:])
3166         if vx < len(gsh.iValues) {
3167             vstr = gsh.iValues[vx]
3168         }else{
3169             }
3170     }
3171     // should use Eval()
3172     if strBegins(vstr, "0x") {
3173         n, err = fmt.Sscanf(vstr[2:], "%x", &ival)
3174     }else{
3175         n, err = fmt.Sscanf(vstr, "%d", &ival)
3176     }
3177     //fmt.Printf("--D-- n=%d err=(%v) (%s)=%v\n", n, err, vstr, ival)
3178     if n == 1 && err == nil {
3179         //fmt.Printf("--D-- formatn(%v) ival(%v)\n", fmts, ival)
3180         fmt.Printf("%"+fmts, ival)
3181     }else{
3182         if isin("-bn", optv){
3183             fmt.Printf("%"+fmts, filepath.Base(vstr))
3184         }else{
3185             fmt.Printf("%"+fmts, vstr)
3186         }
3187     }
3188 }
3189 func (gsh*GshContext)printfv(fmts, div string, argv []string, optv []string, list []string){
3190     //fmt.Printf("(%d)", len(list))
3191     //curfmt := "v"
3192     outlen := 0
3193     curfmt := gsh.iFormat
3194
3195     if 0 < len(fmts) {
3196         for xi := 0; xi < len(fmts); xi++ {
3197             fch := fmts[xi]
3198             if fch == '%' {
3199                 if xi+1 < len(fmts) {
3200                     curfmt = string(fmts[xi+1])
3201                 }
3202                 xi += 1
3203                 if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3204                     vals, leng := scanUntil(fmts[xi+2:], ")")
3205                     //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n", curfmt, vals, leng)
3206                     gsh.printVal(curfmt, vals, optv)
3207                     xi += 2+leng-1
3208                     outlen += 1
3209                 }
3210                 continue
3211             }
3212             if fch == '.' {
3213                 hi, leng := scanInt(fmts[xi+1:])
3214                 if 0 < leng {
3215                     if hi < len(gsh.iValues) {
3216                         gsh.printVal(curfmt, gsh.iValues[hi], optv)
3217                         outlen += 1 // should be the real length
3218                     }else{
3219                         fmt.Printf("(out-range)")
3220                     }
3221                     xi += leng
3222                     continue;
3223                 }
3224             }

```

```

3225     }
3226     fmt.Printf("%c",fch)
3227     outlen += 1
3228 }
3229 }else{
3230 //fmt.Printf("--D-- print {%s}\n")
3231 for i,v := range list {
3232     if 0 < i {
3233         fmt.Printf(div)
3234     }
3235     gsh.printVal(curfmt,v,optv)
3236     outlen += 1
3237 }
3238 }
3239 if 0 < outlen {
3240     fmt.Printf("\n")
3241 }
3242 }
3243 func (gsh*GshContext)Scanv(argv[]string){
3244 //fmt.Printf("--D-- Scanv(%v)\n",argv)
3245 if len(argv) == 1 {
3246     return
3247 }
3248 argv = argv[1:]
3249 fmts := ""
3250 if strBegins(argv[0],"-F") {
3251     fmts = argv[0]
3252     gsh.iDelimiter = fmts
3253     argv = argv[1:]
3254 }
3255 input := strings.Join(argv, " ")
3256 if fmts == "" { // simple decomposition
3257     v := scanv(input)
3258     gsh.iValues = v
3259 //fmt.Printf("%v\n",strings.Join(v,","))
3260 }else{
3261     v := make([]string,0)
3262     n,err := fmt.Sscanf(input,fmts,&v[0],&v[1],&v[2],&v[3])
3263     fmt.Printf("--D-- Sscanf ->(%v) n=%d err=(%v)\n",v,n,err)
3264     gsh.iValues = v
3265 }
3266 }
3267 func (gsh*GshContext)Printv(argv[]string){
3268 if false { //@0U
3269     fmt.Printf("%v\n",strings.Join(argv[1:], " "))
3270     return
3271 }
3272 //fmt.Printf("--D-- Printv(%v)\n",argv)
3273 //fmt.Printf("%v\n",strings.Join(gsh.iValues,","))
3274 div := gsh.iDelimiter
3275 fmts := ""
3276 argv = argv[1:]
3277 if 0 < len(argv) {
3278     if strBegins(argv[0],"-F") {
3279         div = argv[0][2:]
3280         argv = argv[1:]
3281     }
3282 }
3283 }
3284 optv := []string{}
3285 for _,v := range argv {
3286     if strBegins(v,"-"){
3287         optv = append(optv,v)
3288         argv = argv[1:]
3289     }else{
3290         break;
3291     }
3292 }
3293 if 0 < len(argv) {
3294     fmts = strings.Join(argv, " ")
3295 }
3296 gsh.printfv(fmts,div,argv,optv,gsh.iValues)
3297 }
3298 func (gsh*GshContext)Basename(argv[]string){
3299 for i,v := range gsh.iValues {
3300     gsh.iValues[i] = filepath.Base(v)
3301 }
3302 }
3303 func (gsh*GshContext)Sortv(argv[]string){
3304 sv := gsh.iValues
3305 sort.Slice(sv , func(i,j int) bool {
3306     return sv[i] < sv[j]
3307 })
3308 }
3309 func (gsh*GshContext)Shiftv(argv[]string){
3310 vi := len(gsh.iValues)
3311 if 0 < vi {
3312     if isin("-r",argv) {
3313         top := gsh.iValues[0]
3314         gsh.iValues = append(gsh.iValues[1:],top)
3315     }else{
3316         gsh.iValues = gsh.iValues[1:]
3317     }
3318 }
3319 }
3320 }
3321 func (gsh*GshContext)Eng(argv[]string){
3322 }
3323 func (gsh*GshContext)Deq(argv[]string){
3324 }
3325 func (gsh*GshContext)Push(argv[]string){
3326 gsh.iValStack = append(gsh.iValStack,argv[1:])
3327 fmt.Printf("depth=%d\n",len(gsh.iValStack))
3328 }
3329 func (gsh*GshContext)Dump(argv[]string){
3330 for i,v := range gsh.iValStack {
3331     fmt.Printf("%d %v\n",i,v)
3332 }
3333 }
3334 func (gsh*GshContext)Pop(argv[]string){
3335 depth := len(gsh.iValStack)
3336 if 0 < depth {
3337     v := gsh.iValStack[depth-1]
3338     if isin("-cat",argv){
3339         gsh.iValues = append(gsh.iValues,v...)
3340     }else{
3341         gsh.iValues = v
3342     }
3343     gsh.iValStack = gsh.iValStack[0:depth-1]
3344     fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
3345 }else{
3346     fmt.Printf("depth=%d\n",depth)
3347 }
3348 }

```

```

3349
3350 // <a name="interpreter">Command Interpreter</a>
3351 func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
3352     fin = false
3353
3354     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
3355     if len(argv) <= 0 {
3356         return false
3357     }
3358     xargv := []string{}
3359     for ai := 0; ai < len(argv); ai++ {
3360         xargv = append(xargv, strsubst(gshCtx,argv[ai],false))
3361     }
3362     argv = xargv
3363     if false {
3364         for ai := 0; ai < len(argv); ai++ {
3365             fmt.Printf("[%d] %s [%d]T\n",
3366                 ai,argv[ai],len(argv[ai]),argv[ai])
3367         }
3368     }
3369     cmd := argv[0]
3370     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\v\n",len(argv),argv) }
3371     switch { // https://tour.golang.org/flowcontrol/11
3372     case cmd == "":
3373         gshCtx.xPwD([]string{}); // empty command
3374     case cmd == "-x":
3375         gshCtx.CmdTrace = ! gshCtx.CmdTrace
3376     case cmd == "-xt":
3377         gshCtx.CmdTime = ! gshCtx.CmdTime
3378     case cmd == "-ot":
3379         gshCtx.sconnect(true, argv)
3380     case cmd == "-ou":
3381         gshCtx.sconnect(false, argv)
3382     case cmd == "-it":
3383         gshCtx.saccept(true, argv)
3384     case cmd == "-iu":
3385         gshCtx.saccept(false, argv)
3386     case cmd == "-i" || cmd == "<" || cmd == "-o" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
3387         gshCtx.redirect(argv)
3388     case cmd == "|":
3389         gshCtx.fromPipe(argv)
3390     case cmd == "args":
3391         gshCtx.Args(argv)
3392     case cmd == "bg" || cmd == "-bg":
3393         rfin := gshCtx.inBackground(argv[1:])
3394         return rfin
3395     case cmd == "-bn":
3396         gshCtx.Basename(argv)
3397     case cmd == "call":
3398         _,_ = gshCtx.excommand(false,argv[1:])
3399     case cmd == "cd" || cmd == "chdir":
3400         gshCtx.xChdir(argv);
3401     case cmd == "-cksum":
3402         gshCtx.xFind(argv)
3403     case cmd == "-sum":
3404         gshCtx.xFind(argv)
3405     case cmd == "-sumtest":
3406         str := ""
3407         if 1 < len(argv) { str = argv[1] }
3408         crc := strCRC32(str,uint64(len(str)))
3409         fprintf(stderr,"%v %v\n",crc,len(str))
3410     case cmd == "close":
3411         gshCtx.xClose(argv)
3412     case cmd == "gcp":
3413         gshCtx.FileCopy(argv)
3414     case cmd == "dec" || cmd == "decode":
3415         gshCtx.Dec(argv)
3416     case cmd == "#define":
3417     case cmd == "dic" || cmd == "d":
3418         xDic(argv)
3419     case cmd == "dump":
3420         gshCtx.Dump(argv)
3421     case cmd == "echo" || cmd == "e":
3422         echo(argv,true)
3423     case cmd == "enc" || cmd == "encode":
3424         gshCtx.Enc(argv)
3425     case cmd == "env":
3426         env(argv)
3427     case cmd == "eval":
3428         xEval(argv[1:],true)
3429     case cmd == "ev" || cmd == "events":
3430         dumpEvents(argv)
3431     case cmd == "exec":
3432         _,_ = gshCtx.excommand(true,argv[1:])
3433         // should not return here
3434     case cmd == "exit" || cmd == "quit":
3435         // write Result code EXIT to 3>
3436         return true
3437     case cmd == "fds":
3438         // dump the attributes of fds (of other process)
3439     case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
3440         gshCtx.xFind(argv[1:])
3441     case cmd == "fu":
3442         gshCtx.xFind(argv[1:])
3443     case cmd == "fork":
3444         // mainly for a server
3445     case cmd == "-gen":
3446         gshCtx.gen(argv)
3447     case cmd == "-go":
3448         gshCtx.xGo(argv)
3449     case cmd == "-grep":
3450         gshCtx.xFind(argv)
3451     case cmd == "gdeq":
3452         gshCtx.Deq(argv)
3453     case cmd == "genq":
3454         gshCtx.Enq(argv)
3455     case cmd == "gpop":
3456         gshCtx.Pop(argv)
3457     case cmd == "gpush":
3458         gshCtx.Push(argv)
3459     case cmd == "history" || cmd == "hi": // hi should be alias
3460         gshCtx.xHistory(argv)
3461     case cmd == "jobs":
3462         gshCtx.xJobs(argv)
3463     case cmd == "lnsp" || cmd == "nlsp":
3464         gshCtx.SplitLine(argv)
3465     case cmd == "-ls":
3466         gshCtx.xFind(argv)
3467     case cmd == "nop":
3468         // do nothing
3469     case cmd == "pipe":
3470         gshCtx.xOpen(argv)
3471     case cmd == "plug" || cmd == "plugin" || cmd == "pin":
3472         gshCtx.xPlugin(argv[1:])

```

```

3473 case cmd == "print" || cmd == "-pr":
3474     // output internal slice // also sprintf should be
3475     gshCtx.Printv(argv)
3476 case cmd == "ps":
3477     gshCtx.xPs(argv)
3478 case cmd == "pstitle":
3479     // to be gsh.title
3480 case cmd == "rexecc" || cmd == "rexd":
3481     gshCtx.RexecServer(argv)
3482 case cmd == "rexec" || cmd == "rex":
3483     gshCtx.RexecClient(argv)
3484 case cmd == "repeat" || cmd == "rep": // repeat cond command
3485     gshCtx.repeat(argv)
3486 case cmd == "replay":
3487     gshCtx.xReplay(argv)
3488 case cmd == "scan":
3489     // scan input (or so in fscanf) to internal slice (like Files or map)
3490     gshCtx.Scanv(argv)
3491 case cmd == "set":
3492     // set name ...
3493 case cmd == "serv":
3494     gshCtx.httpServer(argv)
3495 case cmd == "shift":
3496     gshCtx.Shiftv(argv)
3497 case cmd == "sleep":
3498     gshCtx.sleep(argv)
3499 case cmd == "-sort":
3500     gshCtx.Sortv(argv)
3501
3502 case cmd == "j" || cmd == "join":
3503     gshCtx.Rjoin(argv)
3504 case cmd == "a" || cmd == "alpa":
3505     gshCtx.Rexec(argv)
3506 case cmd == "jcd" || cmd == "jchdir":
3507     gshCtx.Rchdir(argv)
3508 case cmd == "jget":
3509     gshCtx.Rget(argv)
3510 case cmd == "jls":
3511     gshCtx.Rls(argv)
3512 case cmd == "jput":
3513     gshCtx.Rput(argv)
3514 case cmd == "jpwd":
3515     gshCtx.Rpwd(argv)
3516
3517 case cmd == "time":
3518     fin = gshCtx.xTime(argv)
3519 case cmd == "ungets":
3520     if 1 < len(argv) {
3521         ungets(argv[1]+"\\n")
3522     }else{
3523     }
3524 case cmd == "pwd":
3525     gshCtx.xPwd(argv);
3526 case cmd == "ver" || cmd == "-ver" || cmd == "version":
3527     gshCtx.showVersion(argv)
3528 case cmd == "where":
3529     // data file or so?
3530 case cmd == "which":
3531     which("PATH", argv);
3532 case cmd == "gj" && 1 < len(argv) && argv[1] == "listen":
3533     go gj_server(argv[1:]);
3534 case cmd == "gj" && 1 < len(argv) && argv[1] == "serve":
3535     go gj_server(argv[1:]);
3536 case cmd == "gj" && 1 < len(argv) && argv[1] == "join":
3537     go gj_client(argv[1:]);
3538 case cmd == "gj":
3539     jsend(argv);
3540 case cmd == "jsend":
3541     jsend(argv);
3542 default:
3543     if gshCtx.whichPlugin(cmd,[]string{"-s"}) != nil {
3544         gshCtx.xPlugin(argv)
3545     }else{
3546         notfound, _ := gshCtx.excommand(false, argv)
3547         if notfound {
3548             fmt.Printf("--E-- command not found (%v)\\n",cmd)
3549         }
3550     }
3551 }
3552 return fin
3553 }
3554
3555 func (gsh*GshContext)gshelll(gline string) (rfin bool) {
3556     argv := strings.Split(string(gline)," ")
3557     fin := gsh.gshellv(argv)
3558     return fin
3559 }
3560 func (gsh*GshContext)tgshelll(gline string)(xfin bool){
3561     start := time.Now()
3562     fin := gsh.gshelll(gline)
3563     end := time.Now()
3564     elps := end.Sub(start);
3565     if gsh.CmdTime {
3566         fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + "(%d.%09ds)\\n",
3567             elps/1000000000,elps%1000000000)
3568     }
3569     return fin
3570 }
3571 func Ttyid() (int) {
3572     fi, err := os.Stdin.Stat()
3573     if err != nil {
3574         return 0;
3575     }
3576     //fmt.Printf("Stdin: %v Dev=%d\\n",
3577     // fi.Mode(),fi.Mode()&os.ModeDevice)
3578     if (fi.Mode() & os.ModeDevice) != 0 {
3579         stat := syscall.Stat_t{};
3580         err := syscall.Fstat(0,&stat)
3581         if err != nil {
3582             //fmt.Printf("--I-- Stdin: (%v)\\n",err)
3583         }else{
3584             //fmt.Printf("--I-- Stdin: rdev=%d %d\\n",
3585             // stat.Rdev&0xFF,stat.Rdev);
3586             //fmt.Printf("--I-- Stdin: tty%d\\n",stat.Rdev&0xFF);
3587             return int(stat.Rdev & 0xFF)
3588         }
3589     }
3590     return 0
3591 }
3592 func (gshCtx *GshContext) ttyfile() string {
3593     //fmt.Printf("--I-- GSH_HOME=%s\\n",gshCtx.GshHomeDir)
3594     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
3595         fmt.Sprintf("%02d",gshCtx.TerminalId)
3596     //strconv.Itoa(gshCtx.TerminalId)

```

```

3597 //fmt.Printf("--I-- ttyfile=%s\n",ttyfile)
3598 return ttyfile
3599 }
3600 func (gshCtx *GshContext) ttyline>(*os.File){
3601 file, err := os.OpenFile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
3602 if err != nil {
3603     fmt.Printf("--F-- cannot open %s (%s)\n",gshCtx.ttyfile(),err)
3604     return file;
3605 }
3606 return file
3607 }
3608 func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string) {
3609     if( skipping ){
3610         reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
3611         line, _, _ := reader.ReadLine()
3612         return string(line)
3613     }else
3614     if true {
3615         return xgetline(hix,prevline,gshCtx)
3616     }
3617     /*
3618     else
3619     if( with_exgetline && gshCtx.GetLine != "" ){
3620         //var xhix int64 = int64(hix); // cast
3621         newenv := os.Environ()
3622         newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )
3623
3624         tty := gshCtx.ttyline()
3625         tty.WriteString(prevline)
3626         Pa := os.ProcAttr {
3627             "", // start dir
3628             newenv, //os.Environ(),
3629             []*os.File{os.Stdin,os.Stdout,os.Stderr,tty},
3630             nil,
3631         }
3632         //fmt.Printf("--I-- getline=%s // %s\n",gsh_getlinev[0],gshCtx.GetLine)
3633         proc, err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&Pa)
3634         if err != nil {
3635             fmt.Printf("--F-- getline process error (%v)\n",err)
3636             // for ; ; { }
3637             return "exit (getline program failed)"
3638         }
3639         //stat, err := proc.Wait()
3640         proc.Wait()
3641         buff := make([]byte,LINESIZE)
3642         count, err := tty.Read(buff)
3643         //_, err = tty.Read(buff)
3644         //fmt.Printf("--D-- getline (%d)\n",count)
3645         if err != nil {
3646             if ! (count == 0) { // && err.String() == "EOF" } {
3647                 fmt.Printf("--E-- getline error (%s)\n",err)
3648             }
3649         }else{
3650             //fmt.Printf("--I-- getline OK \"%s\"\n",buff)
3651         }
3652         tty.Close()
3653         gline := string(buff[0:count])
3654         return gline
3655     }else
3656     /*
3657     {
3658         // if isatty {
3659         fmt.Printf("!%d",hix)
3660         fmt.Print(PROMPT)
3661         // }
3662         reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
3663         line, _, _ := reader.ReadLine()
3664         return string(line)
3665     }
3666 }
3667
3668 //== begin ===== getline
3669 /*
3670 * getline.c
3671 * 2020-0819 extracted from dog.c
3672 * getline.go
3673 * 2020-0822 ported to Go
3674 */
3675 /*
3676 package main // getline main
3677 import (
3678     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
3679     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
3680     "os" // <a href="https://golang.org/pkg/os/">os</a>
3681     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
3682     //"bytes" // <a href="https://golang.org/pkg/os/">os</a>
3683     //"os/exec" // <a href="https://golang.org/pkg/os/">os</a>
3684 )
3685 */
3686
3687 // C language compatibility functions
3688 var errno = 0
3689 var stdin *os.File = os.Stdin
3690 var stdout *os.File = os.Stdout
3691 var stderr *os.File = os.Stderr
3692 var EOF = -1
3693 var NULL = 0
3694 type FILE os.File
3695 type StrBuff []byte
3696 var NULL_FP *os.File = nil
3697 var NULLSP = 0
3698 //var LINESIZE = 1024
3699
3700 func system(cmdstr string)(int){
3701     PA := syscall.ProcAttr {
3702         "" // the starting directory
3703         os.Environ(),
3704         []uintptr{os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()},
3705         nil,
3706     }
3707     argv := strings.Split(cmdstr," ")
3708     pid,err := syscall.ForkExec(argv[0],argv,&PA)
3709     if( err != nil ){
3710         fmt.Printf("--E-- syscall(%v) err(%v)\n",cmdstr,err)
3711     }
3712     syscall.Wait4(pid,nil,0,nil)
3713
3714     /*
3715     argv := strings.Split(cmdstr," ")
3716     fmt.Fprintf(os.Stderr,"--I-- system(%v)\n",argv)
3717     //cmd := exec.Command(argv[0]:...)
3718     cmd := exec.Command(argv[0],argv[1],argv[2])
3719     cmd.Stdin = strings.NewReader("output of system")
3720     var out bytes.Buffer

```

```

3721 cmd.Stdout = &out
3722 var serr bytes.Buffer
3723 cmd.Stderr = &serr
3724 err := cmd.Run()
3725 if err != nil {
3726     fmt.Fprintf(os.Stderr, "--E-- system(%v)err(%v)\n", argv, err)
3727     fmt.Printf("ERR:%s\n", serr.String())
3728 }else{
3729     fmt.Printf("%s", out.String())
3730 }
3731 */
3732 return 0
3733 }
3734 func atoi(str string)(ret int){
3735     ret, err := fmt.Sscanf(str, "%d", &ret)
3736     if err == nil {
3737         return ret
3738     }else{
3739         // should set errno
3740         return 0
3741     }
3742 }
3743 func getenv(name string)(string){
3744     val, got := os.LookupEnv(name)
3745     if got {
3746         return val
3747     }else{
3748         return "?"
3749     }
3750 }
3751 func strcpy(dst StrBuff, src string){
3752     var i int
3753     srcb := []byte(src)
3754     for i = 0; i < len(src) && srcb[i] != 0; i++ {
3755         dst[i] = srcb[i]
3756     }
3757     dst[i] = 0
3758 }
3759 func xstrcpy(dst StrBuff, src StrBuff){
3760     dst = src
3761 }
3762 func strcat(dst StrBuff, src StrBuff){
3763     dst = append(dst, src...)
3764 }
3765 func strdup(str StrBuff)(string){
3766     return string(str[0:strlen(str)])
3767 }
3768 func strlen(str string)(int){
3769     return len(str)
3770 }
3771 func strlen(str StrBuff)(int){
3772     var i int
3773     for i = 0; i < len(str) && str[i] != 0; i++ {
3774     }
3775     return i
3776 }
3777 func sizeof(data StrBuff)(int){
3778     return len(data)
3779 }
3780 func isatty(fd int)(ret int){
3781     return 1
3782 }
3783 }
3784 func fopen(file string, mode string)(fp*os.File){
3785     if mode == "r" {
3786         fp, err := os.Open(file)
3787         if( err != nil ){
3788             fmt.Printf("--E-- fopen(%s,%s)=(%v)\n", file, mode, err)
3789             return NULL_FP;
3790         }
3791         return fp;
3792     }else{
3793         fp, err := os.OpenFile(file, os.O_RDWR|os.O_CREATE|os.O_TRUNC, 0600)
3794         if( err != nil ){
3795             return NULL_FP;
3796         }
3797         return fp;
3798     }
3799 }
3800 func fclose(fp*os.File){
3801     fp.Close()
3802 }
3803 func fflush(fp *os.File)(int){
3804     return 0
3805 }
3806 func fgetc(fp*os.File)(int){
3807     var buf [1]byte
3808     _, err := fp.Read(buf[0:1])
3809     if( err != nil ){
3810         return EOF;
3811     }else{
3812         return int(buf[0])
3813     }
3814 }
3815 func sfgets(str*string, size int, fp*os.File)(int){
3816     buf := make(StrBuff, size)
3817     var ch int
3818     var i int
3819     for i = 0; i < len(buf)-1; i++ {
3820         ch = fgetc(fp)
3821         //fprintf(stderr, "--fgets %d/%d %X\n", i, len(buf), ch)
3822         if( ch == EOF ){
3823             break;
3824         }
3825         buf[i] = byte(ch);
3826         if( ch == '\n' ){
3827             break;
3828         }
3829     }
3830     buf[i] = 0
3831     //fprintf(stderr, "--fgets %d/%d (%s)\n", i, len(buf), buf[0:i])
3832     return i
3833 }
3834 func fgets(buf StrBuff, size int, fp*os.File)(int){
3835     var ch int
3836     var i int
3837     for i = 0; i < len(buf)-1; i++ {
3838         ch = fgetc(fp)
3839         //fprintf(stderr, "--fgets %d/%d %X\n", i, len(buf), ch)
3840         if( ch == EOF ){
3841             break;
3842         }
3843         buf[i] = byte(ch);
3844         if( ch == '\n' ){

```

```

3845         break;
3846     }
3847 }
3848 buf[i] = 0
3849 //fprintf(stderr, "--fgets %d/%d (%s)\n", i, len(buf), buf[0:i])
3850 return i
3851 }
3852 func fputc(ch int, fp*os.File)(int){
3853     var buf [1]byte
3854     buf[0] = byte(ch)
3855     fp.Write(buf[0:1])
3856     return 0
3857 }
3858 func fputs(buf StrBuff, fp*os.File)(int){
3859     fp.Write(buf)
3860     return 0
3861 }
3862 func xfprintf(str string, fp*os.File)(int){
3863     return fputs([]byte(str), fp)
3864 }
3865 func sscanf(str StrBuff, fmts string, params ...interface{})(int){
3866     fmt.Sscanf(string(str[0:strlen(str)]), fmts, params...)
3867     return 0
3868 }
3869 func fprintf(fp*os.File, fmts string, params ...interface{})(int){
3870     fmt.Fprintf(fp, fmts, params...)
3871     return 0
3872 }
3873 }
3874 // <a name="IME">Command Line IME</a>
3875 //----- MyIME
3876 var MyIMEVER = "MyIME/0.0.2";
3877 type RomKana struct {
3878     dic string // dictionary ID
3879     pat string // input pattern
3880     out string // output pattern
3881     hit int64 // count of hit and used
3882 }
3883 var dicents = 0
3884 var romkana [1024]RomKana
3885 var Romkan []RomKana
3886 }
3887 func isinDic(str string)(int){
3888     for i, v := range Romkan {
3889         if v.pat == str {
3890             return i
3891         }
3892     }
3893     return -1
3894 }
3895 const (
3896     DIC_COM_LOAD = "im"
3897     DIC_COM_DUMP = "s"
3898     DIC_COM_LIST = "ls"
3899     DIC_COM_ENA = "en"
3900     DIC_COM_DIS = "di"
3901 )
3902 func helpDic(argv []string){
3903     out := stderr
3904     cmd := ""
3905     if 0 < len(argv) { cmd = argv[0] }
3906     fprintf(out, "-- %v Usage\n", cmd)
3907     fprintf(out, "... Commands\n")
3908     fprintf(out, "... %v %v [dicName] [dicURL] -- Import dictionary\n", cmd, DIC_COM_LOAD)
3909     fprintf(out, "... %v %v [pattern] -- Search in dictionary\n", cmd, DIC_COM_DUMP)
3910     fprintf(out, "... %v %v [dicName] -- List dictionaries\n", cmd, DIC_COM_LIST)
3911     fprintf(out, "... %v %v [dicName] -- Disable dictionaries\n", cmd, DIC_COM_DIS)
3912     fprintf(out, "... %v %v [dicName] -- Enable dictionaries\n", cmd, DIC_COM_ENA)
3913     fprintf(out, "... Keys ... %v\n", "ESC can be used for '\\')
3914     fprintf(out, "... \\c -- Reverse the case of the last character\n",)
3915     fprintf(out, "... \\i -- Replace input with translated text\n",)
3916     fprintf(out, "... \\j -- On/Off translation mode\n",)
3917     fprintf(out, "... \\l -- Force Lower Case\n",)
3918     fprintf(out, "... \\u -- Force Upper Case (software CapsLock)\n",)
3919     fprintf(out, "... \\v -- Show translation actions\n",)
3920     fprintf(out, "... \\x -- Replace the last input character with it Hexa-Decimal\n",)
3921 }
3922 func xDic(argv []string){
3923     if len(argv) <= 1 {
3924         helpDic(argv)
3925         return
3926     }
3927     argv = argv[1:]
3928     var debug = false
3929     var info = false
3930     var silent = false
3931     var dump = false
3932     var builtin = false
3933     cmd := argv[0]
3934     argv = argv[1:]
3935     opt := ""
3936     arg := ""
3937 }
3938 if 0 < len(argv) {
3939     arg1 := argv[0]
3940     if arg1[0] == '-' {
3941         switch arg1 {
3942             default: fmt.Printf("--Ed-- Unknown option(%v)\n", arg1)
3943                 return
3944             case "-b": builtin = true
3945                 case "-d": debug = true
3946                 case "-s": silent = true
3947                 case "-v": info = true
3948         }
3949         opt = arg1
3950         argv = argv[1:]
3951     }
3952 }
3953 }
3954 dicName := ""
3955 dicURL := ""
3956 if 0 < len(argv) {
3957     arg = argv[0]
3958     dicName = arg
3959     argv = argv[1:]
3960 }
3961 if 0 < len(argv) {
3962     dicURL = argv[0]
3963     argv = argv[1:]
3964 }
3965 if false {
3966     fprintf(stderr, "--Dd-- com(%v) opt(%v) arg(%v)\n", cmd, opt, arg)
3967 }
3968 if cmd == DIC_COM_LOAD {

```



```

4093 }
4094 }
4095 func loadDefaultDic(dic int){
4096     if( 0 < len(Romkan) ){
4097         return
4098     }
4099     //fprintf(stderr, "\r\n")
4100     xDic([]string{"dic", DIC_COM_LOAD});
4101
4102     var info = false
4103     if info {
4104         fprintf(stderr, "--Id-- Conguraturations!! WorldDic is now activated.\r\n")
4105         fprintf(stderr, "--Id-- enter \"dic\" command for help.\r\n")
4106     }
4107 }
4108 func readDic()(int){
4109     /*
4110     var rk *os.File;
4111     var dic = "MyIME-dic.txt";
4112     //rk = fopen("romkana.txt", "r");
4113     //rk = fopen("JK-JA-morse-dic.txt", "r");
4114     rk = fopen(dic, "r");
4115     if( rk == NULL_FP ){
4116         if( true ){
4117             fprintf(stderr, "--%s-- Could not load %s\n", MyIMEVER, dic);
4118         }
4119         return -1;
4120     }
4121     if( true ){
4122         var di int;
4123         var line = make(StrBuff, 1024);
4124         var pat string
4125         var out string
4126         for di = 0; di < 1024; di++ {
4127             if( fgets(line, sizeof(line), rk) == NULLSP ){
4128                 break;
4129             }
4130             fmt.Sscanf(string(line[0:strlen(line)]), "%s %s", &pat, &out);
4131             //sscanf(line, "%s %s", &pat, &out);
4132             romkana[di].pat = pat;
4133             romkana[di].out = out;
4134             //fprintf(stderr, "--Dd- %s\n", pat, out)
4135         }
4136         dicents += di
4137         if( false ){
4138             fprintf(stderr, "--%s-- loaded romkana.txt [%d]\n", MyIMEVER, di);
4139             for di = 0; di < dicents; di++ {
4140                 fprintf(stderr,
4141                     "%s %s\n", romkana[di].pat, romkana[di].out);
4142             }
4143         }
4144     }
4145     fclose(rk);
4146
4147     //romkana[dicents].pat = "//ddump"
4148     //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
4149     */
4150     return 0;
4151 }
4152 func matchlen(stri string, pati string)(int){
4153     if strBegins(stri, pati) {
4154         return len(pati)
4155     }else{
4156         return 0
4157     }
4158 }
4159 func convs(src string)(string){
4160     var si int;
4161     var sx = len(src);
4162     var di int;
4163     var mi int;
4164     var dstb []byte
4165
4166     for si = 0; si < sx; { // search max. match from the position
4167         if strBegins(src[si:], "%x/") {
4168             // %x/integer/ // s/a/b/
4169             ix := strings.Index(src[si+3:], "/")
4170             if 0 < ix {
4171                 var iv int = 0
4172                 //fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
4173                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4174                 sval := fmt.Sprintf("%x", iv)
4175                 bval := []byte(sval)
4176                 dstb = append(dstb, bval...)
4177                 si = si+3+ix+1
4178                 continue
4179             }
4180         }
4181         if strBegins(src[si:], "%d/") {
4182             // %d/integer/ // s/a/b/
4183             ix := strings.Index(src[si+3:], "/")
4184             if 0 < ix {
4185                 var iv int = 0
4186                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4187                 sval := fmt.Sprintf("%d", iv)
4188                 bval := []byte(sval)
4189                 dstb = append(dstb, bval...)
4190                 si = si+3+ix+1
4191                 continue
4192             }
4193         }
4194         if strBegins(src[si:], "%t") {
4195             now := time.Now()
4196             if true {
4197                 date := now.Format(time.Stamp)
4198                 dstb = append(dstb, []byte(date)...)
4199                 si = si+3
4200             }
4201             continue
4202         }
4203     }
4204     var maxlen int = 0;
4205     var len int;
4206     mi = -1;
4207     for di = 0; di < dicents; di++ {
4208         len = matchlen(src[si:], romkana[di].pat);
4209         if( maxlen < len ){
4210             maxlen = len;
4211             mi = di;
4212         }
4213     }
4214     if( 0 < maxlen ){
4215         out := romkana[mi].out;
4216         dstb = append(dstb, []byte(out)...);
4217         si += maxlen;

```

```

4217     }else{
4218         dstb = append(dstb,src[si])
4219         si += 1;
4220     }
4221 }
4222 return string(dstb)
4223 }
4224 func trans(src string)(int){
4225     dst := convs(src);
4226     xfprintf(dst,stderr);
4227     return 0;
4228 }
4229
4230 //----- LINEEDIT
4231 // "?" at the top of the line means searching history
4232
4233 // should be compatilbe with Telnet
4234 const (
4235     EV_MODE      = 255
4236     EV_IDLE      = 254
4237     EV_TIMEOUT   = 253
4238
4239     GO_UP        = 252 // k
4240     GO_DOWN      = 251 // j
4241     GO_RIGHT     = 250 // l
4242     GO_LEFT      = 249 // h
4243     DEL_RIGHT    = 248 // x
4244     GO_TOPL      = 'A'-0x40 // 0
4245     GO_ENDL      = 'E'-0x40 // $
4246
4247     GO_TOPW      = 239 // b
4248     GO_ENDW      = 238 // e
4249     GO_NEXTW     = 237 // w
4250
4251     GO_FORWCH    = 229 // f
4252     GO_PAIRCH    = 228 // %
4253
4254     GO_DEL       = 219 // d
4255
4256     HI_SRCH_FW   = 209 // /
4257     HI_SRCH_BK   = 208 // ?
4258     HI_SRCH_RFW  = 207 // n
4259     HI_SRCH_RBK  = 206 // N
4260 )
4261
4262 // should return number of octets ready to be read immediately
4263 //fprintf(stderr,"\n--Select(%v %v)\n",err,r.Bits[0])
4264
4265
4266 var EventRecvFd = -1 // file descriptor
4267 var EventSendFd = -1
4268 const EventFdOffset = 1000000
4269 const NormalFdOffset = 100
4270
4271 func putEvent(event int, evarg int){
4272     if true {
4273         if EventRecvFd < 0 {
4274             var pv = [int{-1,-1}]
4275             syscall.Pipe(pv)
4276             EventRecvFd = pv[0]
4277             EventSendFd = pv[1]
4278             //fmt.Printf("--De-- EventPipe created[%v,%v]\n",EventRecvFd,EventSendFd)
4279         }
4280     }else{
4281         if EventRecvFd < 0 {
4282             // the document differs from this spec
4283             // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
4284             sv,err := syscall.Socketpair(syscall.AF_UNIX,syscall.SOCK_STREAM,0)
4285             EventRecvFd = sv[0]
4286             EventSendFd = sv[1]
4287             if err != nil {
4288                 fmt.Printf("--De-- EventSock created[%v,%v](%v)\n",
4289                     EventRecvFd,EventSendFd,err)
4290             }
4291         }
4292     }
4293     var buf = [byte{ byte(event)}]
4294     n,err := syscall.Write(EventSendFd,buf)
4295     if err != nil {
4296         fmt.Printf("--De-- putEvent[%v](%3v)(%v %v)\n",EventSendFd,event,n,err)
4297     }
4298 }
4299 func ungets(str string){
4300     for _,ch := range str {
4301         putEvent(int(ch),0)
4302     }
4303 }
4304 func (gsh*GshContext)xReplay(argv[]string){
4305     hix := 0
4306     tempo := 1.0
4307     xtempo := 1.0
4308     repeat := 1
4309
4310     for _,a := range argv { // tempo
4311         if strBegins(a,"x") {
4312             fmt.Sscanf(a[1:],"%f",&xtempo)
4313             tempo = 1 / xtempo
4314             //fprintf(stderr,"--Dr-- tempo=[%v]%v\n",a[2:],tempo);
4315         }else
4316         if strBegins(a,"r") { // repeat
4317             fmt.Sscanf(a[1:],"%v",&repeat)
4318         }else
4319         if strBegins(a,"!") {
4320             fmt.Sscanf(a[1:],"%d",&hix)
4321         }else{
4322             fmt.Sscanf(a,"%d",&hix)
4323         }
4324     }
4325     if hix == 0 || len(argv) <= 1 {
4326         hix = len(gsh.CommandHistory)-1
4327     }
4328     fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n",hix,xtempo,repeat)
4329     //dumpEvents(hix)
4330     //gsh.xScanReplay(hix,false,repeat,tempo,argv)
4331     go gsh.xScanReplay(hix,true,repeat,tempo,argv)
4332 }
4333
4334 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4335 // 2020-0827 GShell-0.2.3
4336 /*
4337 func FpollIn1(fp *os.File,usec int)(uintptr){
4338     nfd := 1
4339
4340     rdv := syscall.FdSet {}

```

```

4341 fd1 := fp.Fd()
4342 bank1 := fd1/32
4343 mask1 := int32(1 << fd1)
4344 rdv.Bits[bank1] = mask1
4345
4346 fd2 := -1
4347 bank2 := -1
4348 var mask2 int32 = 0
4349
4350 if 0 <= EventRecvFd {
4351     fd2 = EventRecvFd
4352     nfd = fd2 + 1
4353     bank2 = fd2/32
4354     mask2 = int32(1 << fd2)
4355     rdv.Bits[bank2] |= mask2
4356     //fmt.Printf("--De-- EventPoll mask added [%d][%v][%v]\n",fd2,bank2,mask2)
4357 }
4358
4359 tout := syscall.NsecToTimeval(int64(usec*1000))
4360 //n,err := syscall.Select(nfd,&rdv,nil,nil,&stout) // spec. mismatch
4361 err := syscall.Select(nfd,&rdv,nil,nil,&stout)
4362 if err != nil {
4363     //fmt.Printf("--De-- select() err(%v)\n",err)
4364 }
4365 if err == nil {
4366     if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4367         if false {
4368             fmt.Printf("--De-- got Event\n")
4369         }
4370         return uintptr(EventFdOffset + fd2)
4371     }else
4372     if (rdv.Bits[bank1] & mask1) != 0 {
4373         return uintptr(NormalFdOffset + fd1)
4374     }else{
4375         return 1
4376     }
4377 }else{
4378     return 0
4379 }
4380 }
4381 */
4382 func fgetcTimeout1(fp *os.File,usec int)(int){
4383     READ1:
4384     //readyFd := FpollIn1(fp,usec)
4385     readyFd := CFpollIn1(fp,usec)
4386     if readyFd < 100 {
4387         return EV_TIMEOUT
4388     }
4389
4390     var buf [1]byte
4391
4392     if EventFdOffset <= readyFd {
4393         fd := int(readyFd-EventFdOffset)
4394         _,err := syscall.Read(fd,buf[0:1])
4395         if( err != nil ){
4396             return EOF;
4397         }else{
4398             if buf[0] == EV_MODE {
4399                 recvEvent(fd)
4400                 goto READ1
4401             }
4402             return int(buf[0])
4403         }
4404     }
4405
4406     _,err := fp.Read(buf[0:1])
4407     if( err != nil ){
4408         return EOF;
4409     }else{
4410         return int(buf[0])
4411     }
4412 }
4413
4414 func visibleChar(ch int)(string){
4415     switch {
4416     case '!' <= ch && ch <= '-':
4417         return string(ch)
4418     }
4419     switch ch {
4420     case '\': return "\\s"
4421     case '\n': return "\\n"
4422     case '\r': return "\\r"
4423     case '\t': return "\\t"
4424     }
4425     switch ch {
4426     case 0x00: return "NUL"
4427     case 0x07: return "BEL"
4428     case 0x08: return "BS"
4429     case 0x0E: return "SO"
4430     case 0x0F: return "SI"
4431     case 0x1B: return "ESC"
4432     case 0x7F: return "DEL"
4433     }
4434     switch ch {
4435     case EV_IDLE: return fmt.Sprintf("IDLE")
4436     case EV_MODE: return fmt.Sprintf("MODE")
4437     }
4438     return fmt.Sprintf("%X",ch)
4439 }
4440 func recvEvent(fd int){
4441     var buf = make([]byte,1)
4442     _,_ = syscall.Read(fd,buf[0:1])
4443     if( buf[0] != 0 ){
4444         romkanmode = true
4445     }else{
4446         romkanmode = false
4447     }
4448 }
4449 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[string]){
4450     var Start time.Time
4451     var events = []Event{}
4452     for _,e := range Events {
4453         if hix == 0 || e.CmdIndex == hix {
4454             events = append(events,e)
4455         }
4456     }
4457     elen := len(events)
4458     if 0 < elen {
4459         if events[elen-1].event == EV_IDLE {
4460             events = events[0:elen-1]
4461         }
4462     }
4463     for r := 0; r < repeat; r++ {
4464         for i,e := range events {

```

```

4465     nano := e.when.Nanosecond()
4466     micro := nano / 1000
4467     if Start.Second() == 0 {
4468         Start = time.Now()
4469     }
4470     diff := time.Now().Sub(Start)
4471     if replay {
4472         if e.event != EV_IDLE {
4473             putEvent(e.event,0)
4474             if e.event == EV_MODE { // event with arg
4475                 putEvent(int(e.evarg),0)
4476             }
4477         }
4478     }else{
4479         fmt.Printf("%7.3fms %#-3v !%-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",
4480             float64(diff)/1000000.0,
4481             i,
4482             e.CmdIndex,
4483             e.when.Format(time.Stamp),micro,
4484             e.event,e.event,visibleChar(e.event),
4485             float64(e.evarg)/1000000.0)
4486     }
4487     if e.event == EV_IDLE {
4488         d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
4489         //nsleep(time.Duration(e.evarg))
4490         nsleep(d)
4491     }
4492 }
4493 }
4494 }
4495 func dumpEvents(arg[]string){
4496     hix := 0
4497     if l < len(arg) {
4498         fmt.Sscanf(arg[1],"%d",&hix)
4499     }
4500     for i,e := range Events {
4501         nano := e.when.Nanosecond()
4502         micro := nano / 1000
4503         //if e.event != EV_TIMEOUT {
4504         if hix == 0 || e.CmdIndex == hix {
4505             fmt.Printf("#%-3v !%-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",i,
4506                 e.CmdIndex,
4507                 e.when.Format(time.Stamp),micro,
4508                 e.event,e.event,visibleChar(e.event),float64(e.evarg)/1000000.0)
4509         }
4510         //}
4511     }
4512 }
4513 func fgetcTimeout(fp *os.File,usec int)(int){
4514     ch := fgetcTimeout1(fp,usec)
4515     if ch != EV_TIMEOUT {
4516         now := time.Now()
4517         if 0 < len(Events) {
4518             last := Events[len(Events)-1]
4519             dura := int64(now.Sub(last.when))
4520             Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
4521         }
4522         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
4523     }
4524     return ch
4525 }
4526 }
4527 var AtConsoleLineTop = true
4528 var TtyMaxCol = 72 // to be obtained by ioctl?
4529 var EscTimeout = (100*1000)
4530 var (
4531     MODE_VicMode    bool // vi compatible command mode
4532     MODE_ShowMode  bool // shown translation mode, the mode to be retained
4533     romkanmode     bool // shown translation mode, the mode to be retained
4534     MODE_Recursive bool // recursive translation
4535     MODE_CapsLock  bool // software CapsLock
4536     MODE_LowerLock bool // force lower-case character lock
4537     MODE_ViInsert  int // visible insert mode, should be like "I" icon in X Window
4538     MODE_ViTrace   bool // output newline before translation
4539 )
4540 type IInput struct {
4541     lno int
4542     lastlno int
4543     pch []int // input queue
4544     prompt string
4545     line string
4546     right string
4547     inJmode bool
4548     pinJmode bool
4549     waitingMeta string // waiting meta character
4550     LastCmd string
4551 }
4552 func (iin*IInput)Getc(timeoutUs int)(int){
4553     ch1 := EOF
4554     ch2 := EOF
4555     ch3 := EOF
4556     if( 0 < len(iin.pch) ){ // deQ
4557         ch1 = iin.pch[0]
4558         iin.pch = iin.pch[1:]
4559     }else{
4560         ch1 = fgetcTimeout(stdin,timeoutUs);
4561     }
4562     if( ch1 == 033 ){ // escape sequence
4563         ch2 = fgetcTimeout(stdin,EscTimeout);
4564         if( ch2 == EV_TIMEOUT ){
4565             }else{
4566                 ch3 = fgetcTimeout(stdin,EscTimeout);
4567                 if( ch3 == EV_TIMEOUT ){
4568                     iin.pch = append(iin.pch,ch2) // enQ
4569                 }else{
4570                     switch( ch2 ){
4571                         default:
4572                             iin.pch = append(iin.pch,ch2) // enQ
4573                             iin.pch = append(iin.pch,ch3) // enQ
4574                         case '[':
4575                             switch( ch3 ){
4576                                 case 'A': ch1 = GO_UP; // ^
4577                                 case 'B': ch1 = GO_DOWN; // v
4578                                 case 'C': ch1 = GO_RIGHT; // >
4579                                 case 'D': ch1 = GO_LEFT; // <
4580                                 case '3':
4581                                     ch4 := fgetcTimeout(stdin,EscTimeout);
4582                                     if( ch4 == '-' ){
4583                                         //fprintf(stderr,"%x%02X %02X %02X\n",ch1,ch2,ch3,ch4);
4584                                         ch1 = DEL_RIGHT
4585                                     }
4586                                 case '\\':
4587                                     //ch4 := fgetcTimeout(stdin,EscTimeout);
4588

```

```

4589         //fprintf(stderr, "y{%02X %02X %02X %02X}\n", ch1, ch2, ch3, ch4);
4590         switch( ch3 ){
4591             case '-': ch1 = DEL_RIGHT
4592             }
4593         }
4594     }
4595 }
4596 }
4597 return ch1
4598 }
4599 func (inn*IInput)clearline(){
4600     var i int
4601     fprintf(stderr, "\r");
4602     // should be ANSI ESC sequence
4603     for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
4604         fputc(' ', os.Stderr);
4605     }
4606     fprintf(stderr, "\r");
4607 }
4608 func (iin*IInput)Redraw(){
4609     redraw(iin, iin.lno, iin.line, iin.right)
4610 }
4611 func redraw(iin *IInput, lno int, line string, right string){
4612     inMeta := false
4613     showMode := ""
4614     showMeta := "" // visible Meta mode on the cursor position
4615     showLino := fmt.Sprintf("!%d! ", lno)
4616     InsertMark := "" // in visible insert mode
4617
4618     if MODE_VicMode {
4619     }else
4620     if 0 < len(iin.right) {
4621         InsertMark = " "
4622     }
4623
4624     if( 0 < len(iin.waitingMeta) ){
4625         inMeta = true
4626         if iin.waitingMeta[0] != 033 {
4627             showMeta = iin.waitingMeta
4628         }
4629     }
4630     if( romkanmode ){
4631         //romkanmark = " *";
4632     }else{
4633         //romkanmark = "";
4634     }
4635     if MODE_ShowMode {
4636         romkan := "--"
4637         inmeta := "--"
4638         inveri := ""
4639         if MODE_CapsLock {
4640             inmeta = "A"
4641         }
4642         if MODE_LowerLock {
4643             inmeta = "a"
4644         }
4645         if MODE_ViTrace {
4646             inveri = "v"
4647         }
4648         if MODE_VicMode {
4649             inveri = ":"
4650         }
4651         if romkanmode {
4652             romkan = "\343\201\202"
4653             if MODE_CapsLock {
4654                 inmeta = "R"
4655             }else{
4656                 inmeta = "r"
4657             }
4658         }
4659         if inMeta {
4660             inmeta = "\\ "
4661         }
4662         showMode = "["+romkan+inmeta+inveri+"]";
4663     }
4664     Pre := "\r" + showMode + showLino
4665     Output := ""
4666     Left := ""
4667     Right := ""
4668     if romkanmode {
4669         Left = convs(line)
4670         Right = InsertMark+convs(right)
4671     }else{
4672         Left = line
4673         Right = InsertMark+right
4674     }
4675     Output = Pre+Left
4676     if MODE_ViTrace {
4677         Output += iin.LastCmd
4678     }
4679     Output += showMeta+Right
4680     for len(Output) < TtyMaxCol { // to the max. position that may be dirty
4681         Output += " "
4682         // should be ANSI ESC sequence
4683         // not necessary just after newline
4684     }
4685     Output += Pre+Left+showMeta // to set the cursor to the current input position
4686     fprintf(stderr, "%s", Output)
4687
4688     if MODE_ViTrace {
4689         if 0 < len(iin.LastCmd) {
4690             iin.LastCmd = ""
4691             fprintf(stderr, "\r\n")
4692         }
4693     }
4694     AtConsoleLineTop = false
4695 }
4696 // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
4697 func delHeadChar(str string)(rline string, head string){
4698     _, clen := utf8.DecodeRune([]byte(str))
4699     head = string(str[0:clen])
4700     return str[clen:], head
4701 }
4702 func delTailChar(str string)(rline string, last string){
4703     var i = 0
4704     var clen = 0
4705     for {
4706         _, siz := utf8.DecodeRune([]byte(str)[i:])
4707         if siz <= 0 { break }
4708         clen = siz
4709         i += siz
4710     }
4711     last = str[len(str)-clen:]
4712     return str[0:len(str)-clen:], last

```

```

4713 }
4714 // 3> for output and history
4715 // 4> for keylog?
4716 // <a name="getline">Command Line Editor</a>
4717 func xgetline(lno int, prevline string, gsh*GshContext)(string){
4718     var iin IInput
4719     iin.lastlno = lno
4720     iin.lno = lno
4721
4722     CmdIndex = len(gsh.CommandHistory)
4723     if( isatty(0) == 0 ){
4724         if( sfgets(&iin.line,LINESIZE,stdin) == NULL ){
4725             iin.line = "exit\n";
4726         }else{
4727             return iin.line
4728         }
4729     }
4730     if( true ){
4731         //var pts string;
4732         //pts = ptsname(0);
4733         //pts = ttyname(0);
4734         //fprintf(stderr,"--pts[0] = %s\n",pts?pts:"?");
4735     }
4736     if( false ){
4737         fprintf(stderr,"! ");
4738         fflush(stderr);
4739         sfgets(&iin.line,LINESIZE,stdin);
4740         return iin.line
4741     }
4742     system("/bin/stty -echo -icanon");
4743     xline := iin.xgetline1(prevline,gsh)
4744     system("/bin/stty echo sane");
4745     return xline
4746 }
4747 func (iin*IInput)Translate(cmdch int){
4748     romkanmode = !romkanmode;
4749     if MODE_ViTrace {
4750         fprintf(stderr,"%v\r\n",string(cmdch));
4751     }else
4752     if( cmdch == 'J' ){
4753         fprintf(stderr,"J\r\n");
4754         iin.inJmode = true
4755     }
4756     iin.Redraw();
4757     loadDefaultDic(cmdch);
4758     iin.Redraw();
4759 }
4760 func (iin*IInput)Replace(cmdch int){
4761     iin.LastCmd = fmt.Sprintf("%v",string(cmdch))
4762     iin.Redraw();
4763     loadDefaultDic(cmdch);
4764     dst := convs(iin.line+iin.right);
4765     iin.line = dst
4766     iin.right = ""
4767     if( cmdch == 'I' ){
4768         fprintf(stderr,"I\r\n");
4769         iin.inJmode = true
4770     }
4771     iin.Redraw();
4772 }
4773 // aa 12 alal
4774 func isAlpha(ch rune)(bool){
4775     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
4776         return true
4777     }
4778     return false
4779 }
4780 func isAlnum(ch rune)(bool){
4781     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
4782         return true
4783     }
4784     if '0' <= ch && ch <= '9' {
4785         return true
4786     }
4787     return false
4788 }
4789 // 0.2.8 2020-0901 created
4790 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
4791 func (iin*IInput)GotoTOPW(){
4792     str := iin.line
4793     i := len(str)
4794     if i <= 0 {
4795         return
4796     }
4797     //i0 := i
4798     i -= 1
4799     lastSize := 0
4800     var lastRune rune
4801     var found = -1
4802     for 0 < i { // skip preamble spaces
4803         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4804         if !isAlnum(lastRune) { // character, type, or string to be searched
4805             i -= lastSize
4806             continue
4807         }
4808         break
4809     }
4810     for 0 < i {
4811         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4812         if lastSize <= 0 { continue } // not the character top
4813         if !isAlnum(lastRune) { // character, type, or string to be searched
4814             found = i
4815             break
4816         }
4817         i -= lastSize
4818     }
4819     if found < 0 && i == 0 {
4820         found = 0
4821     }
4822     if 0 <= found {
4823         if isAlnum(lastRune) { // or non-kana character
4824             // when positioning to the top o the word
4825             i += lastSize
4826         }
4827         iin.right = str[i:] + iin.right
4828         if 0 < i {
4829             iin.line = str[0:i]
4830         }else{
4831             iin.line = ""
4832         }
4833     }
4834     //fmt.Printf("\n(%d,%d,%d)[%s][%s]\n",i0,i,found,iin.line,iin.right)
4835 }

```

```

4837 //fmt.Printf("") // set debug messae at the end of line
4838 }
4839 // 0.2.8 2020-0901 created
4840 func (iin*Input)GotoENDW(){
4841     str := iin.right
4842     if len(str) <= 0 {
4843         return
4844     }
4845     lastSize := 0
4846     var lastRune rune
4847     var lastW = 0
4848     i := 0
4849     inWord := false
4850
4851     lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
4852     if isAlnum(lastRune) {
4853         r,z := utf8.DecodeRuneInString(str[lastSize:])
4854         if 0 < z && isAlnum(r) {
4855             inWord = true
4856         }
4857     }
4858     for i < len(str) {
4859         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4860         if lastSize <= 0 { break } // broken data?
4861         if !isAlnum(lastRune) { // character, type, or string to be searched
4862             break
4863         }
4864         lastW = i // the last alnum if in alnum word
4865         i += lastSize
4866     }
4867     if inWord {
4868         goto DISP
4869     }
4870     for i < len(str) {
4871         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4872         if lastSize <= 0 { break } // broken data?
4873         if isAlnum(lastRune) { // character, type, or string to be searched
4874             break
4875         }
4876         i += lastSize
4877     }
4878     for i < len(str) {
4879         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4880         if lastSize <= 0 { break } // broken data?
4881         if !isAlnum(lastRune) { // character, type, or string to be searched
4882             break
4883         }
4884         lastW = i
4885         i += lastSize
4886     }
4887 DISP:
4888     if 0 < lastW {
4889         iin.line = iin.line + str[0:lastW]
4890         iin.right = str[lastW:]
4891     }
4892     //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
4893     //fmt.Printf("") // set debug messae at the end of line
4894 }
4895 // 0.2.8 2020-0901 created
4896 func (iin*Input)GotoNEXTW(){
4897     str := iin.right
4898     if len(str) <= 0 {
4899         return
4900     }
4901     lastSize := 0
4902     var lastRune rune
4903     var found = -1
4904     i := 1
4905     for i < len(str) {
4906         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4907         if lastSize <= 0 { break } // broken data?
4908         if !isAlnum(lastRune) { // character, type, or string to be searched
4909             found = i
4910             break
4911         }
4912         i += lastSize
4913     }
4914     if 0 < found {
4915         if isAlnum(lastRune) { // or non-kana character
4916             }else{ // when positioning to the top o the word
4917                 found += lastSize
4918             }
4919             iin.line = iin.line + str[0:found]
4920             if 0 < found {
4921                 iin.right = str[found:]
4922             }else{
4923                 iin.right = ""
4924             }
4925         }
4926         //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
4927         //fmt.Printf("") // set debug messae at the end of line
4928     }
4929 // 0.2.8 2020-0902 created
4930 func (iin*Input)GotoPAIRCH(){
4931     str := iin.right
4932     if len(str) <= 0 {
4933         return
4934     }
4935     lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
4936     if lastSize <= 0 {
4937         return
4938     }
4939     forw := false
4940     back := false
4941     pair := ""
4942     switch string(lastRune){
4943     case "(": pair = ")"; forw = true
4944     case ")": pair = "("; back = true
4945     case "{": pair = "}"; forw = true
4946     case "}": pair = "{"; back = true
4947     case "[": pair = "]"; forw = true
4948     case "]": pair = "["; back = true
4949     case "<": pair = ">"; forw = true
4950     case ">": pair = "<"; back = true
4951     case "\"": pair = "\""; // context depednet, can be f" or back-double quote
4952     case "'": pair = "'"; // context depednet, can be f' or back-quote
4953     // case Japanese Kakkos
4954     }
4955     if forw {
4956         iin.SearchForward(pair)
4957     }
4958     if back {
4959         iin.SearchBackward(pair)
4960     }

```



```

4961 }
4962 // 0.2.8 2020-0902 created
4963 func (iin*IInput)SearchForward(pat string)(bool){
4964     right := iin.right
4965     found := -1
4966     i := 0
4967     if strBegins(right,pat) {
4968         _,z := utf8.DecodeRuneInString(right[i:])
4969         if 0 < z {
4970             i += z
4971         }
4972     }
4973     for i < len(right) {
4974         if strBegins(right[i:],pat) {
4975             found = i
4976             break
4977         }
4978         _,z := utf8.DecodeRuneInString(right[i:])
4979         if z <= 0 { break }
4980         i += z
4981     }
4982     if 0 <= found {
4983         iin.line = iin.line + right[0:found]
4984         iin.right = iin.right[found:]
4985         return true
4986     }else{
4987         return false
4988     }
4989 }
4990 // 0.2.8 2020-0902 created
4991 func (iin*IInput)SearchBackward(pat string)(bool){
4992     line := iin.line
4993     found := -1
4994     i := len(line)-1
4995     for i = i; 0 <= i; i-- {
4996         _,z := utf8.DecodeRuneInString(line[i:])
4997         if z <= 0 {
4998             continue
4999         }
5000         //fprintf(stderr,"-- %v %v\n",pat,line[i:])
5001         if strBegins(line[i:],pat) {
5002             found = i
5003             break
5004         }
5005     }
5006     //fprintf(stderr,"--%d\n",found)
5007     if 0 <= found {
5008         iin.right = line[found:] + iin.right
5009         iin.line = line[0:found]
5010         return true
5011     }else{
5012         return false
5013     }
5014 }
5015 // 0.2.8 2020-0902 created
5016 // search from top, end, or current position
5017 func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool,string){
5018     if forw {
5019         for _,v := range gsh.CommandHistory {
5020             if 0 <= strings.Index(v.CmdLine,pat) {
5021                 //fprintf(stderr,"\n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
5022                 return true,v.CmdLine
5023             }
5024         }
5025     }else{
5026         hlen := len(gsh.CommandHistory)
5027         for i := hlen-1; 0 < i; i-- {
5028             v := gsh.CommandHistory[i]
5029             if 0 <= strings.Index(v.CmdLine,pat) {
5030                 //fprintf(stderr,"\n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
5031                 return true,v.CmdLine
5032             }
5033         }
5034     }
5035     //fprintf(stderr,"\n--De-- not-found(%v)\n",pat)
5036     return false,"(Not Found in History)"
5037 }
5038 // 0.2.8 2020-0902 created
5039 func (iin*IInput)GotoFORWSTR(pat string,gsh*GshContext){
5040     found := false
5041     if 0 < len(iin.right) {
5042         found = iin.SearchForward(pat)
5043     }
5044     if !found {
5045         found,line := gsh.SearchHistory(pat,true)
5046         if found {
5047             iin.line = line
5048             iin.right = ""
5049         }
5050     }
5051 }
5052 func (iin*IInput)GotoBACKSTR(pat string, gsh*GshContext){
5053     found := false
5054     if 0 < len(iin.line) {
5055         found = iin.SearchBackward(pat)
5056     }
5057     if !found {
5058         found,line := gsh.SearchHistory(pat,false)
5059         if found {
5060             iin.line = line
5061             iin.right = ""
5062         }
5063     }
5064 }
5065 func (iin*IInput)getStringl(prompt string)(string){ // should be editable
5066     iin.clearline();
5067     fprintf(stderr,"\r%",prompt)
5068     str := ""
5069     for {
5070         ch := iin.Getc(10*1000*1000)
5071         if ch == '\n' || ch == '\r' {
5072             break
5073         }
5074         sch := string(ch)
5075         str += sch
5076         fprintf(stderr,"%s",sch)
5077     }
5078     return str
5079 }
5080
5081 // search pattern must be an array and selectable with ^N/^P
5082 var SearchPat = ""
5083 var SearchForw = true
5084

```

```

5085 func (iin*IInput)xgetline(prevline string, gsh*GshContext)(string){
5086     var ch int;
5087
5088     MODE_ShowMode = false
5089     MODE_VicMode = false
5090     iin.Redraw();
5091     first := true
5092
5093     for cix := 0; ; cix++ {
5094         iin.pinJmode = iin.inJmode
5095         iin.inJmode = false
5096
5097         ch = iin.Getc(1000*1000)
5098
5099         if ch != EV_TIMEOUT && first {
5100             first = false
5101             mode := 0
5102             if romkanmode {
5103                 mode = 1
5104             }
5105             now := time.Now()
5106             Events = append(Events,Event{now,EV_MODE,int64(mode),CmdIndex})
5107         }
5108         if ch == 033 {
5109             MODE_ShowMode = true
5110             MODE_VicMode = !MODE_VicMode
5111             iin.Redraw();
5112             continue
5113         }
5114         if MODE_VicMode {
5115             switch ch {
5116                 case '0': ch = GO_TOPL
5117                 case '$': ch = GO_ENDL
5118                 case 'b': ch = GO_TOPW
5119                 case 'e': ch = GO_ENDW
5120                 case 'w': ch = GO_NEXTW
5121                 case '%': ch = GO_PAIRCH
5122
5123                 case 'j': ch = GO_DOWN
5124                 case 'k': ch = GO_UP
5125                 case 'h': ch = GO_LEFT
5126                 case 'l': ch = GO_RIGHT
5127                 case 'x': ch = DEL_RIGHT
5128                 case 'a': MODE_VicMode = !MODE_VicMode
5129                     ch = GO_RIGHT
5130                 case 'i': MODE_VicMode = !MODE_VicMode
5131                     iin.Redraw();
5132                     continue
5133                 case '-':
5134                     right,head := delHeadChar(iin.right)
5135                     if len([]byte(head)) == 1 {
5136                         ch = int(head[0])
5137                         if( 'a' <= ch && ch <= 'z' ){
5138                             ch = ch + 'A'-'a'
5139                         }else
5140                         if( 'A' <= ch && ch <= 'Z' ){
5141                             ch = ch + 'a'-'A'
5142                         }
5143                         iin.right = string(ch) + right
5144                     }
5145                     iin.Redraw();
5146                     continue
5147                 case 'f': // GO_FORWCH
5148                     iin.Redraw();
5149                     ch = iin.Getc(3*1000*1000)
5150                     if ch == EV_TIMEOUT {
5151                         iin.Redraw();
5152                         continue
5153                     }
5154                     SearchPat = string(ch)
5155                     SearchForw = true
5156                     iin.GotoFORWSTR(SearchPat,gsh)
5157                     iin.Redraw();
5158                     continue
5159                 case '/':
5160                     SearchPat = iin.getstringl("/") // should be editable
5161                     SearchForw = true
5162                     iin.GotoFORWSTR(SearchPat,gsh)
5163                     iin.Redraw();
5164                     continue
5165                 case '?':
5166                     SearchPat = iin.getstringl("?") // should be editable
5167                     SearchForw = false
5168                     iin.GotoBACKSTR(SearchPat,gsh)
5169                     iin.Redraw();
5170                     continue
5171                 case 'n':
5172                     if SearchForw {
5173                         iin.GotoFORWSTR(SearchPat,gsh)
5174                     }else{
5175                         iin.GotoBACKSTR(SearchPat,gsh)
5176                     }
5177                     iin.Redraw();
5178                     continue
5179                 case 'N':
5180                     if !SearchForw {
5181                         iin.GotoFORWSTR(SearchPat,gsh)
5182                     }else{
5183                         iin.GotoBACKSTR(SearchPat,gsh)
5184                     }
5185                     iin.Redraw();
5186                     continue
5187             }
5188         }
5189         switch ch {
5190             case GO_TOPW:
5191                 iin.GotoTOPW()
5192                 iin.Redraw();
5193                 continue
5194             case GO_ENDW:
5195                 iin.GotoENDW()
5196                 iin.Redraw();
5197                 continue
5198             case GO_NEXTW:
5199                 // to next space then
5200                 iin.GotoNEXTW()
5201                 iin.Redraw();
5202                 continue
5203             case GO_PAIRCH:
5204                 iin.GotoPAIRCH()
5205                 iin.Redraw();
5206                 continue
5207         }
5208     }

```

```

5209 //fprintf(stderr, "A[%02X]\n",ch);
5210 if( ch == '\\\ ' || ch == 033 ){
5211     MODE_ShowMode = true
5212     metach := ch
5213     iin.waitingMeta = string(ch)
5214     iin.Redraw();
5215     // set cursor //fprintf(stderr, "???\b\b\b")
5216     ch = fgetcTimeout(stdin,2000*1000)
5217     // reset cursor
5218     iin.waitingMeta = ""
5219
5220     cmdch := ch
5221     if( ch == EV_TIMEOUT ){
5222         if metach == 033 {
5223             continue
5224         }
5225         ch = metach
5226     }else
5227     /*
5228     if( ch == 'm' || ch == 'M' ){
5229         mch := fgetcTimeout(stdin,1000*1000)
5230         if mch == 'r' {
5231             romkanmode = true
5232         }else{
5233             romkanmode = false
5234         }
5235         continue
5236     }else
5237     */
5238     if( ch == 'k' || ch == 'K' ){
5239         MODE_Recursive = !MODE_Recursive
5240         iin.Translate(cmdch);
5241         continue
5242     }else
5243     if( ch == 'j' || ch == 'J' ){
5244         iin.Translate(cmdch);
5245         continue
5246     }else
5247     if( ch == 'i' || ch == 'I' ){
5248         iin.Replace(cmdch);
5249         continue
5250     }else
5251     if( ch == 'l' || ch == 'L' ){
5252         MODE_LowerLock = !MODE_LowerLock
5253         MODE_CapsLock = false
5254         if MODE_ViTrace {
5255             fprintf(stderr, "%v\r\n", string(cmdch));
5256         }
5257         iin.Redraw();
5258         continue
5259     }else
5260     if( ch == 'u' || ch == 'U' ){
5261         MODE_CapsLock = !MODE_CapsLock
5262         MODE_LowerLock = false
5263         if MODE_ViTrace {
5264             fprintf(stderr, "%v\r\n", string(cmdch));
5265         }
5266         iin.Redraw();
5267         continue
5268     }else
5269     if( ch == 'v' || ch == 'V' ){
5270         MODE_ViTrace = !MODE_ViTrace
5271         if MODE_ViTrace {
5272             fprintf(stderr, "%v\r\n", string(cmdch));
5273         }
5274         iin.Redraw();
5275         continue
5276     }else
5277     if( ch == 'c' || ch == 'C' ){
5278         if 0 < len(iin.line) {
5279             xline,tail := delTailChar(iin.line)
5280             if len([]byte(tail)) == 1 {
5281                 ch = int(tail[0])
5282                 if 'a' <= ch && ch <= 'z' ){
5283                     ch = ch + 'A'-'a'
5284                 }else
5285                 if 'A' <= ch && ch <= 'Z' ){
5286                     ch = ch + 'a'-'A'
5287                 }
5288                 iin.line = xline + string(ch)
5289             }
5290         }
5291         if MODE_ViTrace {
5292             fprintf(stderr, "%v\r\n", string(cmdch));
5293         }
5294         iin.Redraw();
5295         continue
5296     }else{
5297         iin.pch = append(iin.pch,ch) // push
5298         ch = '\\\ '
5299     }
5300 }
5301 switch( ch ){
5302 case 'P'-0x40: ch = GO_UP
5303 case 'N'-0x40: ch = GO_DOWN
5304 case 'B'-0x40: ch = GO_LEFT
5305 case 'F'-0x40: ch = GO_RIGHT
5306 }
5307 //fprintf(stderr, "B[%02X]\n",ch);
5308 switch( ch ){
5309 case 0:
5310     continue;
5311
5312 case '\t':
5313     iin.Replace('j');
5314     continue
5315 case 'X'-0x40:
5316     iin.Replace('j');
5317     continue
5318
5319 case EV_TIMEOUT:
5320     iin.Redraw();
5321     if iin.pinJmode {
5322         fprintf(stderr, "\\J\r\n")
5323         iin.inJmode = true
5324     }
5325     continue
5326 case GO_UP:
5327     if iin.lno == 1 {
5328         continue
5329     }
5330     cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
5331     if ok {
5332         iin.line = cmd

```

```

5333     iin.right = ""
5334     iin.lno = iin.lno - 1
5335 }
5336 iin.Redraw();
5337 continue
5338 case GO_DOWN:
5339     cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
5340     if ok {
5341         iin.line = cmd
5342         iin.right = ""
5343         iin.lno = iin.lno + 1
5344     }else{
5345         iin.line = ""
5346         iin.right = ""
5347         if iin.lno == iin.lastlno-1 {
5348             iin.lno = iin.lno + 1
5349         }
5350     }
5351     iin.Redraw();
5352     continue
5353 case GO_LEFT:
5354     if 0 < len(iin.line) {
5355         xline,tail := delTailChar(iin.line)
5356         iin.line = xline
5357         iin.right = tail + iin.right
5358     }
5359     iin.Redraw();
5360     continue;
5361 case GO_RIGHT:
5362     if( 0 < len(iin.right) && iin.right[0] != 0 ){
5363         xright,head := delHeadChar(iin.right)
5364         iin.right = xright
5365         iin.line += head
5366     }
5367     iin.Redraw();
5368     continue;
5369 case EOF:
5370     goto EXIT;
5371 case 'R'-0x40: // replace
5372     dst := convs(iin.line+iin.right);
5373     iin.line = dst
5374     iin.right = ""
5375     iin.Redraw();
5376     continue;
5377 case 'T'-0x40: // just show the result
5378     readDic();
5379     romkanmode = !romkanmode;
5380     iin.Redraw();
5381     continue;
5382 case 'L'-0x40:
5383     iin.Redraw();
5384     continue
5385 case 'K'-0x40:
5386     iin.right = ""
5387     iin.Redraw();
5388     continue
5389 case 'E'-0x40:
5390     iin.line += iin.right
5391     iin.right = ""
5392     iin.Redraw();
5393     continue
5394 case 'A'-0x40:
5395     iin.right = iin.line + iin.right
5396     iin.line = ""
5397     iin.Redraw();
5398     continue
5399 case 'U'-0x40:
5400     iin.line = ""
5401     iin.right = ""
5402     iin.clearLine();
5403     iin.Redraw();
5404     continue;
5405 case DEL_RIGHT:
5406     if( 0 < len(iin.right) ){
5407         iin.right,_ = delHeadChar(iin.right)
5408         iin.Redraw();
5409     }
5410     continue;
5411 case 0x7F: // BS? not DEL
5412     if( 0 < len(iin.line) ){
5413         iin.line,_ = delTailChar(iin.line)
5414         iin.Redraw();
5415     }
5416     /*
5417     else
5418         if( 0 < len(iin.right) ){
5419             iin.right,_ = delHeadChar(iin.right)
5420             iin.Redraw();
5421         }
5422     */
5423     continue;
5424 case 'H'-0x40:
5425     if( 0 < len(iin.line) ){
5426         iin.line,_ = delTailChar(iin.line)
5427         iin.Redraw();
5428     }
5429     continue;
5430 }
5431 if( ch == '\n' || ch == '\r' ){
5432     iin.line += iin.right;
5433     iin.right = ""
5434     iin.Redraw();
5435     fputc(ch,stderr);
5436     AtConsoleLineTop = true
5437     break;
5438 }
5439 if MODE_CapsLock {
5440     if 'a' <= ch && ch <= 'z' {
5441         ch = ch+'A'-'a'
5442     }
5443 }
5444 if MODE_LowerLock {
5445     if 'A' <= ch && ch <= 'Z' {
5446         ch = ch+'a'-'A'
5447     }
5448 }
5449 iin.line += string(ch);
5450 iin.Redraw();
5451 }
5452 EXIT:
5453     return iin.line + iin.right;
5454 }
5455 }
5456 func getline_main(){

```

```

5457 line := xgetline(0, "", nil)
5458 fprintf(stderr, "%s\n", line);
5459 /*
5460 dp = strpbrk(line, "\r\n");
5461 if( dp != NULL ){
5462     *dp = 0;
5463 }
5464
5465 if( 0 ){
5466     fprintf(stderr, "\n(%d)\n", int(strlen(line)));
5467 }
5468 if( lseek(3,0,0) == 0 ){
5469     if( romkanmode ){
5470         var buf [8*1024]byte;
5471         convs(line, buf);
5472         stropy(line, buf);
5473     }
5474     write(3, line, strlen(line));
5475     ftruncate(3, lseek(3,0,SEEK_CUR));
5476     //fprintf(stderr, "outsize=%d\n", (int)lseek(3,0,SEEK_END));
5477     lseek(3,0,SEEK_SET);
5478     close(3);
5479 }else{
5480     fprintf(stderr, "\r\ngotline: ");
5481     trans(line);
5482     //printf("%s\n", line);
5483     printf("\n");
5484 }
5485 */
5486 }
5487 //== end ===== getline
5488
5489 //
5490 // $USERHOME/.gsh/
5491 //     gsh-rc.txt, or gsh-configure.txt
5492 //     gsh-history.txt
5493 //     gsh-aliases.txt // should be conditional?
5494 //
5495 func (gshCtx *GshContext)gshSetupHomedir()(bool) {
5496     homedir, found := userHomeDir()
5497     if !found {
5498         fmt.Printf("--E-- You have no UserHomeDir\n")
5499         return true
5500     }
5501     gshhome := homedir + "/" + GSH_HOME
5502     _, err2 := os.Stat(gshhome)
5503     if err2 != nil {
5504         err3 := os.Mkdir(gshhome, 0700)
5505         if err3 != nil {
5506             fmt.Printf("--E-- Could not Create %s (%s)\n",
5507                 gshhome, err3)
5508             return true
5509         }
5510         fmt.Printf("--I-- Created %s\n", gshhome)
5511     }
5512     gshCtx.GshHomeDir = gshhome
5513     return false
5514 }
5515 func setupGshContext()(GshContext, bool){
5516     gshPA := syscall.ProcAttr {
5517         "", // the staring directory
5518         os.Environ(), // environ[]
5519         []uintptr{os.Stdin.Fd(), os.Stdout.Fd(), os.Stderr.Fd()},
5520         nil, // OS specific
5521     }
5522     cwd, _ := os.Getwd()
5523     gshCtx := GshContext {
5524         cwd, // StartDir
5525         "", // GetLine
5526         []GchdirHistory { {cwd, time.Now(), 0} }, // ChdirHistory
5527         gshPA,
5528         []GCommandHistory {}, // something for invocation?
5529         []GCommandHistory {}, // CmdCurrent
5530         false,
5531         []int {},
5532         syscall.Rusage {},
5533         "", // GshHomeDir
5534         Ttyid(),
5535         false,
5536         false,
5537         []PluginInfo {},
5538         []string {},
5539         "",
5540         "v",
5541         ValueStack {},
5542         GServer {"", ""}, // LastServer
5543         "", // RSERV
5544         cwd, // RWD
5545         CheckSum {},
5546     }
5547     err := gshCtx.gshSetupHomedir()
5548     return gshCtx, err
5549 }
5550 func (gsh *GshContext)gshelllh(gline string)(bool){
5551     ghist := gsh.CmdCurrent
5552     ghist.WorkDir, _ = os.Getwd()
5553     ghist.WorkDirX = len(gsh.ChdirHistory) - 1
5554     //fmt.Printf("--D--ChdirHistory(%d)\n", len(gsh.ChdirHistory))
5555     ghist.StartAt = time.Now()
5556     rusagev1 := Getrusagev()
5557     gsh.CmdCurrent.FoundFile = []string {}
5558     fin := gsh.tgshelll(gline)
5559     rusagev2 := Getrusagev()
5560     ghist.Rusagev = RusageSubv(rusagev2, rusagev1)
5561     ghist.EndAt = time.Now()
5562     ghist.CmdLine = gline
5563     ghist.FoundFile = gsh.CmdCurrent.FoundFile
5564
5565     /* record it but not show in list by default
5566     if len(gline) == 0 {
5567         continue
5568     }
5569     if gline == "hi" || gline == "history" { // don't record it
5570         continue
5571     }
5572     */
5573     gsh.CommandHistory = append(gsh.CommandHistory, ghist)
5574     return fin
5575 }
5576 // <a name="main">Main loop</a>
5577 func script(gshCtxGiven *GshContext) (_ GshContext) {
5578     gshCtxBuf, err0 := setupGshContext()
5579     if err0 {
5580         return gshCtxBuf;

```

```

5581 }
5582 gshCtx := &gshCtxBuf
5583
5584 //fmt.Printf("--I-- GSH_HOME=%s\n", gshCtx.GshHomeDir)
5585 //resmap()
5586
5587 /*
5588 if false {
5589     gsh_getlinev, with_exgetline :=
5590         which("PATH", []string{"which", "gsh-getline", "-s"})
5591     if with_exgetline {
5592         gsh_getlinev[0] = toFullPath(gsh_getlinev[0])
5593         gshCtx.GetLine = toFullPath(gsh_getlinev[0])
5594     }else{
5595         fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
5596     }
5597 }
5598 */
5599
5600 ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
5601 gshCtx.CommandHistory = append(gshCtx.CommandHistory, ghist0)
5602
5603 prevline := ""
5604 skipping := false
5605 for hix := len(gshCtx.CommandHistory); ; {
5606     gline := gshCtx.getline(hix, skipping, prevline)
5607     if skipping {
5608         if strings.Index(gline, "fi") == 0 {
5609             fmt.Printf("fi\n");
5610             skipping = false;
5611         }else{
5612             //fmt.Printf("%s\n", gline);
5613         }
5614         continue
5615     }
5616     if strings.Index(gline, "if") == 0 {
5617         //fmt.Printf("--D-- if start: %s\n", gline);
5618         skipping = true;
5619         continue
5620     }
5621     if false {
5622         os.Stdout.Write([]byte("gotline:"))
5623         os.Stdout.Write([]byte(gline))
5624         os.Stdout.Write([]byte("\n"))
5625     }
5626     gline = strsubst(gshCtx, gline, true)
5627     if false {
5628         fmt.Printf("fmt.Printf %%v - %v\n", gline)
5629         fmt.Printf("fmt.Printf %%s - %s\n", gline)
5630         fmt.Printf("fmt.Printf %%x - %s\n", gline)
5631         fmt.Printf("fmt.Printf %%U - %s\n", gline)
5632         fmt.Printf("Stouut. Write -")
5633         os.Stdout.Write([]byte(gline))
5634         fmt.Printf("\n")
5635     }
5636     /*
5637     // should be cared in substitution ?
5638     if 0 < len(gline) && gline[0] == '!' {
5639         xgline, set, err := searchHistory(gshCtx, gline)
5640         if err {
5641             continue
5642         }
5643         if set {
5644             // set the line in command line editor
5645         }
5646         gline = xgline
5647     }
5648     */
5649     fin := gshCtx.gshellllh(gline)
5650     if fin {
5651         break;
5652     }
5653     prevline = gline;
5654     hix++;
5655 }
5656 return *gshCtx
5657 }
5658 func main() {
5659     gshCtxBuf := GshContext{}
5660     gsh := &gshCtxBuf
5661     argv := os.Args
5662
5663     if( isin("wss", argv) ){
5664         gj_server(argv[1:]);
5665         return;
5666     }
5667     if( isin("wsc", argv) ){
5668         gj_client(argv[1:]);
5669         return;
5670     }
5671     if 1 < len(argv) {
5672         if isin("version", argv){
5673             gsh.showVersion(argv)
5674             return
5675         }
5676         if argv[1] == "gj" {
5677             if argv[2] == "listen" { go gj_server(argv[2:]); }
5678             if argv[2] == "server" { go gj_server(argv[2:]); }
5679             if argv[2] == "serve" { go gj_server(argv[2:]); }
5680             if argv[2] == "client" { go gj_client(argv[2:]); }
5681             if argv[2] == "join" { go gj_client(argv[2:]); }
5682         }
5683         comx := isinX("-c", argv)
5684         if 0 < comx {
5685             gshCtxBuf.err := setupGshContext()
5686             gsh := &gshCtxBuf
5687             if !err {
5688                 gsh.gshellv(argv[comx+1:])
5689             }
5690             return
5691         }
5692     }
5693     if 1 < len(argv) && isin("-s", argv) {
5694     }else{
5695         gsh.showVersion(append(argv, []string{"-l", "-a"}...))
5696     }
5697     script(nil)
5698     //gshCtx := script(nil)
5699     //gshellll(gshCtx, "time")
5700 }
5701
5702 //</div></details>
5703 //<details id="gsh-todo"><summary>Considerations</summary><div class="gsh-src">
5704 // - inter gsh communication, possibly running in remote hosts -- to be remote shell

```



```

5829 <a href="https://golang.org/pkg/">Packages</a>
5830 <a href="https://godoc.org/golang.org/x/net/websocket">WebSocket</a>
5831
5832 <a href="https://stackoverflow.com/">Stackoverflow</a>
5833 <!--
5834 <iframe src="https://golang.org" width="100%" height="300"></iframe>
5835 -->
5836 </div></details>
5837 */
5838 /*
5839 <details id="html-src" onclick="frame_open();"><summary>Raw Source</summary></div>
5840
5841 <!-- h2>The full of this HTML including the Go code is here.</h2 -->
5842 <details id="gsh-whole-view"><summary>Whole file</summary>
5843 <a name="whole-src-view"></a>
5844 <span id="src-frame"></span><!-- a window to show source code -->
5845 </details>
5846
5847 <details id="gsh-style-frame" onclick="fill_CSSView()"><summary>CSS part</summary>
5848 <a name="style-src-view"></a>
5849 <span id="gsh-style-view"></span>
5850 </details>
5851
5852 <details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
5853 <a name="script-src-view"></a>
5854 <span id="gsh-script-view"></span>
5855 </details>
5856
5857 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Builtin data part</summary>
5858 <a name="gsh-data-frame"></a>
5859 <span id="gsh-data-view"></span>
5860 </details>
5861
5862 </div></details>
5863 */
5864 /*
5865 <div id="GshFooter0"></div>
5866 <!-- 2020-09-17 Sat@ITS, visible script { -- >
5867 <details><summary>GJScript</summary>
5868 <style>gjscript { font-family:Georgia; }</style>
5869 <pre id="gjscript_1" class="gjscript"> function gjtest1() { alert('Hello GJScript!'); }
5870 gjtest1()
5871 </pre>
5872 <script>
5873 gjs = document.getElementById('gjscript_1');
5874 //eval(gjs.innerHTML);
5875 //gjs.outerHTML = ""
5876 </script>
5877 </details><!-- ----- END-OF-VISIBLE-PART ----- } -->
5878
5879
5880 <!--
5881 // 2020-0906 added,
5882 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
5883 https://developer.mozilla.org/en-US/docs/Web/CSS/position
5884 -->
5885 <span id="GshGrid">(^_^)</small>{Hit j k l h}</small></span>
5886
5887 <span id="GStat"><br>
5888 </span>
5889 <span id="GMenu" onclick="GShellMenu(this)"></span>
5890 <span id="GTop"></span>
5891 <div id="GShellPlane" onclick="showGShellPlane();"></div>
5892 <div id="RawTextViewer"></div>
5893 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
5894
5895 <style id="GshStyleDef">
5896 #LineNumbered table,tr,td {
5897 margin:0;
5898 padding:4px;
5899 spacing:0;
5900 border:12px;
5901 }
5902 textarea.LineNumber {
5903 font-size:12px;
5904 font-family:monospace,Courier New;
5905 color:#282;
5906 padding:4px;
5907 text-align:right;
5908 }
5909 textarea.LineNumbered {
5910 font-size:12px;
5911 font-family:monospace,Courier New;
5912 padding:4px;
5913 wrap:off;
5914 }
5915 #RawTextViewer{
5916 z-index:0;
5917 position:fixed; top:0px; left:0px;
5918 width:100%; xxxheight:50px; xheight:0px;
5919 overflow:auto;
5920 color:#fff; background-color:rgba(128,128,256,0.2);
5921 font-size:12px;
5922 spellcheck:false;
5923 }
5924 #RawTextViewerClose{
5925 z-index:0;
5926 position:fixed; top:-100px; left:-100px;
5927 color:#fff; background-color:rgba(128,128,256,0.2);
5928 font-size:20px; font-family:Georgia;
5929 white-space:pre;
5930 }
5931 #xxxGShellPlane{
5932 z-index:0;
5933 position:fixed; top:0px; left:0px;
5934 width:100%; height:50px;
5935 overflow:auto;
5936 color:#fff; background-color:rgba(128,128,256,0.3);
5937 font-size:12px;
5938 }
5939 #xxxGTop{
5940 z-index:9;
5941 opacity:1.0;
5942 position:fixed; top:0px; left:0px;
5943 width:320px; height:20px;
5944 color:#fff; background-color:rgba(32,32,160,0.15);
5945 color:#fff; font-size:12px;
5946 }
5947 #xxxGPos{
5948 z-index:12;
5949 position:fixed; top:0px; left:0px;
5950 opacity:1.0;
5951 width:640px; height:30px;
5952 color:#fff; background-color:rgba(0,0,0,0.2);

```



```

5953   color:#fff; font-size:12px;
5954 }
5955 #GMenu{
5956   z-index:2000;
5957   position:fixed; top:250px; left:0px;
5958   opacity:1.0;
5959   width:100px; height:100px;
5960   color:#fff;
5961   color:#fff; background-color:rgba(0,0,0,0.0);
5962   color:#fff; font-size:16px; font-family:Georgia;
5963   background-repeat:no-repeat;
5964 }
5965 #xxxGStat{
5966   z-index:8;
5967   xopacity:0.0;
5968   position:fixed; top:20px; left:0px;
5969   xwidth:640px;
5970   width:100%; height:90px;
5971   color:#fff; background-color:rgba(0,0,128,0.04);
5972   font-size:20px; font-family:Georgia;
5973 }
5974 #GLog{
5975   z-index:10;
5976   position:fixed; top:50px; left:0px;
5977   opacity:1.0;
5978   width:640px; height:60px;
5979   color:#fff; background-color:rgba(0,0,128,0.10);
5980   font-size:12px;
5981 }
5982 #GshGrid {
5983   z-index:11;
5984   xopacity:0.0;
5985   position:fixed; top:0px; left:0px;
5986   width:320px; height:30px;
5987   color:#9f9; font-size:16px;
5988 }
5989 xbody {display:none;}
5990 .gsh-link{color:green;}
5991 #gsh {border-width:1px;margin:0;padding:0;}
5992 #gsh {font-family:monospace,Courier New;color:#ddf;font-size:8px;}
5993 #gsh header{height:100px;}
5994 #xgsh header{height:100px;background-image:url(GShell-Logo00.png);}
5995 #GshMenu{font-size:14pt;color:#c44;}
5996 .GshMenu{
5997   font-size:14pt;color:#2a2;padding:4px; text-align:right;
5998 }
5999 .GshMenu: hover{
6000   font-size:14pt;color:#fff;font-weight:bold;background-color:#2a2;
6001 }
6002 #GshFooter{height:100px;background-size:80px;background-repeat:no-repeat;}
6003 #gsh note{color:#000;font-size:10pt;}
6004 #gsh h2{color:#24a;font-family:Georgia;font-size:18pt;}
6005 #gsh h3{color:#24a;font-family:Georgia;font-size:16pt;}
6006 #gsh details{color:#888;background-color:#fff;font-family:monospace;}
6007 #gsh summary{font-size:16pt;color:#fff;background-color:#8af; height:30px;}
6008 #gsh pre{font-size:11pt;color:#223;background-color:#fafff;}
6009 #gsh a{color:#24a;}
6010 #gsh a{name}{color:#24a;font-size:16pt;}
6011 #gsh .gsh-src{white-space:pre;font-family:monospace,Courier New;font-size:11pt;}
6012 #gsh .gsh-src{background-color:#fafff;color:#223;}
6013 #gsh-src-src{spellcheck:false}
6014 #SrcTextarea{white-space:pre;font-family:Courier New;font-size:10pt;}
6015 #SrcTextarea{background-color:#fafff;color:#223;}
6016 .gsh-code {white-space:pre;font-family:Courier New !important;}
6017 .gsh-code {color:#024;font-size:11pt; background-color:#fafff;}
6018 .gsh-golang-data {display:none;}
6019 #gsh-WinId {color:#000;font-size:14pt;}
6020
6021 .gsh-document {font-size:11pt;background-color:#fff;font-family:Georgia;}
6022 .gsh-document {color:#000;background-color:#fff !important;}
6023 .gsh-document > h2{color:#000;background-color:#fff !important;}
6024 .gsh-document details{color:#000;background-color:#fff;font-family:Georgia;}
6025 .gsh-document p{max-width:550pt;color:#000;background-color:#fff;font-family:Georgia;}
6026 .gsh-document address{width:500pt;color:#000;background-color:#fff;font-family:Georgia;}
6027
6028 @media print {
6029   #gsh pre{font-size:11pt !important;}
6030 }
6031 </style>
6032
6033 <!--
6034 // Logo image should be drawn by JavaScript from a meta-font.
6035 // CSS seems not follow line-splitted URL
6036 -->
6037 <script id="gsh-data">
6038 //GSellLogo="QR-ITS-more.jp.png"
6039 GSellLogo="data:image/png;base64,\
6040 1VBORw0KGgoAAAANSUHUgAAQGAAB/CAYAAADvs3f4AAAAAXNSRO1Ars4c6QAAHh1WElm\
6041 TU0AKgAAAaAgBAAEAAUAAABAAAPgPBAAUAAABAAARgEoAAARAAABAAIAAIdpAAQAAAB\
6042 AAARtPAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAAB\
6043 AAAAQAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAABAAAB\
6044 x++tTukZ3iCcdgO/jy60sb8WgMzAvn7uG4+biSTR7YnQkdQPCKg2j2nWLD2MSLRkEUpnoD\
6045 4iJx7jrjY25D0DGMF2Vq1BEiSggCoiMMA+mu+vu//ZMD9U1da62a2Ubv91Krq3vvd6/q\
6046 fnXvxd8tBAsIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6047 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6048 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6049 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6050 2eXs9H9+ftSkSdHxsic2qgdE7Yus+1qaalKfnY5YsokMHWEPtdK4MQFz5UeExlLbLYsYU15\
6051 mpDIKKEZC1FIRMS3JSUaq9ScqcU6i+2kK3StuOny5reEGKJ7Qw7MvKec2ToqO1zwo1jhFS\
6052 jboVHCstMRb3USXEJ8hFu7DsmdFb2+uU4vWVFXbBpMeZULAE/hcKoGAb66eKGO1Nyhk56PC\
6053 HxH2VVBKORqk3qUeKilYdaOfONJ560kdi6w5BwomOQlyPzi0N9DLmXpFK/60P2P/Piyovf\
6054 n8mf++/nJWngjw9KqToLVGSFt2p2R11gn3i0V7kYsoWVzEuVpF1RKYdfoakZLRSB0q\
6055 zrWocCOG6qEhvgRacJ/dktj3g7dXXH4gKN6aRS0zPzYzergS6RAoZDQqFk79SKTRXHu/+e9FN\
6056 166as88pU/PNlp1TLQJKSc73dPXSR20ur71iWpCc8QhbnCnCyHuilryyOTQvYF5fvgBL7jx\
6057 +cNHjBj5gJrYdLJHY39o84D40H2Qtx8THaPeFuIOU+wlC+KnyhK5FGEV0WGGAEb83eXMoLY\
6058 r1kbb9gHEP25Vg14h89FU66KJyFbbQbnZLJg42FiesndHCwvUosEV00/SC9F911Ue0H\
6059 +zChU9nSgQmouGurK19RjBD4Y6uQeQd5TU0M63zd3MhEsy14V91sbdKf;xbGh1CpFR\
6060 UJ6toACF79V9P58NBFDH0TOMBaE74Ent+eWrr+Lz/Q7w60AdB7QUjps/OA7c00NBCEMUZ\
6061 ttCu/coG28FLpvKE1TPPF8juRasEahbHvxaR1quoeBPyfUD04+OfEbdyB8L4tz9XeSPMOC\
6062 bgGov0v1zgGw4jP392xnHhdC+Mwf3TjTfnt22yC1YJBYXNUt5KIKyck1xSRd1d6mceV\
6063 aJovy/VBacMeVqEP46/ZlnJ7+9jx17VL53215MtVap1QLNhw5pQDqXyNTQ122b8nGcM2V\
6064 q0QfjSjYvV0AZ2Dfayidv6FJ35CS4jXZk9h1r7e27zm6p3T8HJpKJicYjP1HtK/D/DFU4Jw1\
6065 1ImhX5IR9z2zqRkx4w/C+HQSPe+krbIyRn3qEPTNahsHdLs2xh5Q5NcOPVdEgqcbm/8e\
6066 7/dzoAptag/mLKJ77U0VG0xybTdx/Ex/PtA/i7r7Ku+cSoiCuxUrohUxP16wE9H+ccVg1\
6067 pd/CFU42AK2IUp1vTK1L/sjYyE5PVHq728NzvFuzvDODGy9GoopuuhMLNfctX48YHL2QW\
6068 f/8hXVU/43rQ9xtg6Ytvc1XDC3fmWQDn9nbE21e7wKE1b0K651cBuEqhd3IAw82dwKPUw\
6069 hrauc6zcWdcjUZK8EUXMae71zUgWcu2nbi6eVn1J19/P7eW+loMAogP+NI31JLSf8dn9ipA\
6070 WNW4P9yJxUpeDL/HXzNzTsvesLD2vswHWI9mu5rvVzX9foS4V/LfmqDEIPHDGLf2uCW\
6071 gjTy2wENPa23fEcivd+ZYNCNjYrNyhyGAo8jRoJTAUmriqOCJnRW5fPTN++frTwdh4SiUw\
6072 bVLWbVfLcRF04qazRD7176/rBjKyLD5pBi25Wi4wQu71kPBeCOpUw+Kj0sqP8GNhAZuwl\
6073 10zuywDh9zBr2x0DRqVQF15QxxH6owVjRKAAW46pvt+RxAJVLjW7vY9/+CEUMK168/rPQn\
6074 mCuftZvldfN/yI8gA5iW3dkIKhsvy2ucYSVG/KhchwFWRDKAMMcD8EKX+rHF12A9bt2d172\
6075 2gNzov2CDYmfEtNy7QoqDXWIKAIQ7c0Q2chyADWnnergN5xVxtcJsdGp20twqmJU7A+EH7\
6076 yHbUgm1X7f7K1DwaryUFN42F1uxNDdVETamL6sYC9R26VtbZaW2px8Nfmezh3EM+mgs0Lk\

```

6077 d3/zNBGE1XPQUwzXg1Yc5eW5/zBgy54aW0gWkfnWbptceVWT4FUBvov32gew8DLzDTMaj\
6078 augp7t/bMxk+yw/egJKoTKsy2d+gfBb9VodvX5B12TOR+wfjyBp6U0XGQNGYR/qua3vB\
6079 Fgeue6y2d7vN8dFV3rldBw34G8P910DG955XkNhn9KAbAmMyJ6dK1Pz2mD3Cnu177vtv5\
6080 h/rTc1p7p0/vvurDuc+ws654mm+8zK0yqRSPRA4IKoGz1i8b5ytagcFmb9v/m9oEUATv\
6081 Jow6LVnPCmHzj+sNnHscCyj1a6csarRmMyRkxiw415i0u1i1LRWfmlNLeX3z2+/cfW1LJ2\
6082 Y572b6EAzqYoPctJi15Q1nJyLdxFrU2p1/3pmkug/yn9GaQoYMTf7neV1vx/6CUHgh1lh/\
6083 f9Uvo+gD03q7rZFL8x0+zW+/8F6PW6fV7sXhInLayvWdz2X1Ulm/4u1LmPNoA5oGdOL9\
6084 2FA6goczhTgQ641NR5Doj9xuv1cy+rFbcujVsnLkKv0CEfphUbICLRmV1+9KP4vngH6GfC2\
6085 NCQM5iCsnCkfxEd+mtFlBwuxdmF02qT/194225Y3TCrzpQWhthG2zHraJ0/yb0kdkhpanzQ\
6086 GxwF66/8Cb5AHcbzdpnUjge6YFow1g2emmtGcDekzTiXVuc3Lk4yVTJepuq5tqSWFKXdA\
6087 uFu9MfiWIG3sgnNtcX76+3xEXQWZVeqSpvrZmC2afY5V461+04KvYVgicCugG2rpoYPTveJ\
6088 o2Ulm2JWE2D+f6K0FtNXfV2U9x70/bqzct5z0Po10+vdpYDjcdxR3D4U9XcHrLoSktt3ug\
6089 AcwtK009Ftn+gWtWds60DcFodrAxneOCFRXWUS0K93pB2XM7vAe+gwr506/204IXqngLbrC\
6090 76HgrdveHz2W1MYVvqgm5zTTP5+7v0LRR/zJ10Y1x+h0h0Eeb+CV/0TU5ic3NGfjKs30M2\
6091 LFUt1+Y14yfAcwkjzqpb6HlGJebwpgLXyo09/j8k//WW3xS2gQPhrV5AmTp1IDFN2Op6\
6092 fz5yW4HfMDX+Buy4NVu73yFbP0K651cot+zjP+8qf4JKyTnGKTB/qST0zMKACq18jPGL\
6093 A4PCXyNPKOTrJEv84HpyOsws/BsqyT2RGZ6zr210gA9sBhEP46hsP2ratm0JeGrugWBDB2Pw\
6094 NYD1B40STmcmcdS2E/GG2ZvrF7Uejsqwy/7A7gUeH6Kyy19q3fP0qQvXtX4dz+Ueg+Lmy5v\
6095 bjJyE0+b5Lsqp5Nz6nwbFPhUdaYgemz4ap1z5d1bYvA3MTC4F3RKYfOTkaUF9Xry0LwUw\
6096 sDMC/HZ9v0GTWV1C+i2hTu27rGaEabkb4+8h3P553q00yu/WHJ21Wbd722XLuv44f1gmQSV\
6097 2GML+6m8vrvagWme1lyz/gLLX+IBNcn2FQ7F95XQW/gUa+Hr3UzAgj1MTR435fFyBtP\
6098 me65oyCZcJbXmX9QZ3qgbcBymXSL9Vz08BfXkUBjHpbzbz+hvKueRBR1oE/Bw80gf/LIzH\
6099 /9Tcnsb681L7DtdgnQRE8LEvT2z9eWt58j71F5Z0VlyfTLvqU70b62etccR011Hes68SvE\
6100 20zUdeqWRM7S7ng7dkrV19zLz0MPBK73nA4YrdZfm+5D2symDymaHnC1okvPOVHGSF\
6101 wCY6RwU9Dkx5MU9vQXMA+ePguLw8/dvf6U1LpvsPbpXsp01QwagELsm9gqNxtc0EO1j5\
6102 7TBBjAdHkMpdY0/q/irW1bF44t5cNKQKwq7DsuJzH1C6Lz8bk+1u2u78FYWfkl04/qYz\
6103 LYvJX8boyWm6zwc9/Ojwz7pUvtLp0NQ2UxLo8PKODMuluvoovTDjLycxrNWEHjQWsyKrkPs\
6104 2JH14LpJicXQoy6nMs5fysKeile0G95+Wxcej3mcmjNe5b+lyH2YELXgJrMdnY/HtMk0S\
6105 aPE7Md34PueUyZ8DWdov5zXVF/xsFe+Lpz/wjQQ9eiH94ZwqV62+CuH73LmNjSHfXorP\
6106 wKgz9Fw1rCRJwJwh5+/ocSLzQzG52BvItg+wOpqXRYEwcarfRdbS5Cb/PysXBHakPWO\
6107 qzXy41U0ABB4kx5e8qBdH06+hbnwzjzFXyAuViy6Ece001I7SA2kx0UgmtZB9RcaVyx\
6108 2CMBjdAdTcrUWYKriw4myTH9z3R93/8X1j0ESWetyy7qFI1odwkAmhFEA2K6D16WneH\
6109 H52hWwIaLQHQOYUz76y2nTls7rgu40YBJq4JBWJCayRhtyeY4X8/xCw+rUS9L5yc50A+W\
6110 8v0wN2AxW7VADPzCEDpXpdsLX0KefrEM+yj47aEaa7yXzjXm+61F2U46ch7c0d6Q/m\
6111 Wncf9BTvXbs623hNxpIvm1kjhUbtFKRbaq1QCWibui1PtyKlHwZag8YKoeMcj19Y9Lw\
6112 Pwk79U/55Bk75fSXmchwj79Y35Xy7qu8Y8spvTbqSG+55hdj6Y56ErfyqVOLz6e3r1r9Y\
6113 Twkg5S2p1OK5dJzgs+2L1B426/gG+uoa6yU0W1jzCuoG41lqXQVYep1w1XUL4PPR\
6114 zTGL6v1VEJjA35xepk1NLSubb/3ARcw6JXGz65fFlBbJhJ7LmbGDV8Db4b1KgpPbhW\
6115 M0F31qMh7TEHivuxx458PpQWRNdipfV4Z2b1xhEFl6+rtdGLV82CTVYB1d8e2J7FpwJk\
6116 aW3rXYbgm9qXMLmChjcnvUN5FKMR2LbzJbk8m55c4z/2rLDQzNjtkyuu01pdgcccFMz\
6117 gKp/ahfXooV+JToFimZuJyn87F0HmhAMxAdAeUX6c7F07sUUKgyg50z33v/20C7+sch\
6118 Lt.nplth3Yev84ipGt4JwAnu7Pn5xwqjx841MabBc308rflzPCfTc0SF08Nad2SFwqfHbU\
6119 nmldjITHgh3vSrt+42M5KwCtsxFMe35RJTvorP3rmm49VM0fP8oid191X61dvbXmkj3e\
6120 NfydX9mWim2MLKzEsL/VzQSKDpZcdYcyte7lq/B4XKfQaNeK3M47r29FL/gaT+vrEO\
6121 dGTTX0U9WUKUMVfh9MYuL2jVpZxxu0FP00/pTedh0d/LXXGzawfuXp6EGILz+eme2X9lbo\
6122 0xuU119P0baLgG0hafa5NVPhxjK7X0gLUOmRR+JAFesnnaKzLRhZLYB5ediUwKc1/wD7\
6123 fD+JL72vEtPEIqgWkzj6zFP/d5duzt+2HihxfKlnhs7umT011AjKkyVScenpJ1WALAACzE\
6124 dgV2Sx+S+n1N0dPELXvtD/SkUr+JL5/9svbL75z+bYNS8Q2EuQn/Oa3x1/FJZS/VZ30EGCb\
6125 ePdtCYCRORCKr3q6vL0pof7FXvDAaVzccjQECZ56CYcmxZ/7CyuWar2IIN2x4NOC075/\
6126 4yMTRK3XuyfGjGmxt/xdpbt8uSri7F11luoFJTm3U17cKXfYqVfDvwpV99PaeH07FRv\
6127 hUL4693puwYyN+FX0C+Cy0VrIWXzyLh/w3n7fEiibreUtTsURM1tjpkWRyMFKZmHDZFcii\
6128 dMf1f6+ew10/651MmCDD2YFEL2DFycg38ARabQSPG1sCGUCaCRDOUysauzavgcZ6zAvTF\
6129 LLGqFLPjFjyithCkphr+cN+r76LoLd1d3d451+sndv9Yr4veCWg9+SrxTcx6G/arezLXB4W\
6130 tgzv7Wk4n+Z8f/FFZUK1A3k5U0m9CE8N3HqLln151sRNY32hsXoXnTBmWm1P9zT0a3\
6131 308881G578Ej2N0adq77EY0Z2L8Sg+TC9Z0vUe1j3spxVgH8euc3RkK1W0VSSU1rMhw\
6132 xpJdJm+X0QFruceEUTDVLpFenovWf2KAeYAr65T3+3BQCF+5rCU+U1LzE1P/CJmJWRV\
6133 4Yuz9vP9xLfvw/0puyxDP9uNPyih9L/XNv0NSd4dGG8Cwms3CzrKcQUJCSjijvms8q\
6134 Jv7X3Xm6WqjLSr6LVB6870eXtHj7/4Cwd24+vzFvsJrsT1LrkFO0ALtznF2dJ2S12rQ\
6135 8YV88Dsbv0hRQD6evxR0j9y49DQPkC5Zmjy2011dV7y3zfl8qmsDmOFARVWVFC3D\
6136 N1NQGW1jEavqOmRz78D22VefmcdpfcU8n6BFB5+KFL1FMRHE6F080at0m/d8V8km7D\
6137 C58YrseFLVspLpbx79z64erd2NyuNLKileJdalUak7j0orr315x+YA9CBQBD/cK7JKHdD\
6138 E5sg690KMHpdRjD63vgEvYbdQucuc1VM9n0/QaPP3K2Lve8zCmJ3k30kx+30RQK0E8K1W\
6139 lBxaFhe29JbL8of8GKam6n5P9mdGp5bmUikpmc22r7BHSKjyJp0kMctkCf/KAMLS0JtXejk\
6140 v7g+/OzmZbn/Z5IoHT3+NPgzn2eyx7u1z0ZJDM9xoyTCZBT0ya+vndqW3URP1jYxmbD0e1/\
6141 zq4BrYggs1mphGDILKxcmLwXskzBw940stveb+Sf714iK3oi05mXod+r9/12Vx80P9Ec3\
6142 xp7XQYU8JGtmcatO+NeY/99v3xhb+21bh03cn0t1jdfcZnzkeapSDN/vjDg4XP4Cnb8+W9P\
6143 9z2duKz2Q3fev051ygtqmo30pzk9Ec5shOY+FXF150r6xr58HkDFMGAADQ3yA09Bydfj\
6144 ppf5kjq6qrN13dfyKI5h14oOKj1aZehbJ9NtWTFBAGv1uIawS2xvTahfsB50dfrcpaeEaP\
6145 mRiFLX0m8Xm4xnp/fEby6AvG2f2y5SkWnoZmPESF3sgcF304UGGSj/w1548wLfbvVab720b\
6146 Xx/Mrwf1E9zrXPQMbmX5C1AfjIhIyXjhsR/BKMF8GmlT+D3cJF2qod1vN3V3d60w7hyF\
6147 koSvF0EpkFRegJwTld706cndp1H7z10z933h0LHYW1u1Reh27pctve69XW+3Jdasmt0\
6148 zW5s1G578Ej2N0adq77EY0Z2L8Sg+TC9Z0vUe1j3spxVgH8euc3RkK1W0VSSU1rMhw\
6149 LqyxfpWk1KQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIK\
6150 QAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIK\
6151 QAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIK\
6152 QAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIK\
6153 QAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIK\
6154 QAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIKQAIK\
6155 m38w0cAAAAASUVORK5CYII=";

```

6201 "ZYIhrIaC3tqxsInDCYApDVR0NHbX5sJLW1W8pWnHHR0Bn10Uy/S1gHMI3hb23K4wJ5Mvbus"+
6202 "rxwAe6t9U3Ig+AKvB8uTqUgDNH6g2KomOEjd1buNX9dN4LEp52EVret/IAJ9r9ajGsBJZ1"+
6203 "gOR809DazEFFIPpsxB8kV9z/LrFNtY9AZFLlhb134DAZNPf84c44zdkHeqKk1wR014UN"+
6204 "FVYxOzFupD4sXLcU/HKf69s9Pt99SPuhdgdL2dPhobB8uPFW0aFaz99iy/eUumwvffz4"+
6205 "eFLBs20quzCWTAEJUCok386Vpm2IYyay060a0SoowGqblArzqJULHeg555c5e4k3911te"+
6206 "k/Wt+anFaspPh/yodkGR6DJPkC15tjBSDOTORkMyCnBLOCCar9zvQrda04DjTLDVzjrFL"+
6207 "9mMMRQVx1pSap94NPWwsj4mdz5ieBszRLjmgEcO11hmlZXFAMs0dyKONeTgBLpGHGUz6W"+
6208 "X4F67eKisysy94VLYorkTRo6932be+13JCGDgNPNkHwQ9IdJUElqRcBETjdKevqXVJDV"+
6209 "XDMAMWEPBvpu1LLFsmwSuk3Qcxe2gTFWuzRjHEMy6dyp+TL567yUGagCKfKdb1jKagW9"+
6210 "zV2YskHouLTd/hnjp8yH6zizscwKNJ1K1kIDXM341QOGHAjg1A2nqWdZuh+mfIXASSHEer"+
6211 "QKcCXEYMPVJ70MnCjSON8tdlpBsJzLJodNG4Ej5FkoZ+NtV5VMkLrLn4BMMD+Pfhv10Hu"+
6212 "mXkQwUvUPDDNM1QOXVNG26B0jNB15ca58RSuXhQgi2XjNwG1LlEY2ne+b2tLSR1b14xCuvL"+
6213 "dCAGwDgWxGzq7FU2mOXNfLzgh45F+SYkrZb7MIzUJ9Whf4iAu1N8Y52Qa52IBgo3yG8CO"+
6214 "wjb5KmhKtF6pbz4OXALahvEd8x2daZafhuwEznm3FbKHG0T/DnwFSO/wTwmCoZhv8WgSHSD"+
6215 "L8IPbr9kwlhKVDLav+8j3VID+rUoPvHBXYD+zKMBmbac9z+caefAKU/CR0s/D6yGwL79i2d"+
6216 "eZqwxRYegew3GEa6GfSj7Lbtz2l0pxw2wBCinER21rFvffZqx/P/MdPeJ0+I73s64yy+oWX"+
6217 "f+RHZLViErjgg/7paY9GzFLfbziURK6S00jtb18XGYMORXOPNjYVK7xDCKbrAke11dSAe"+
6218 "jXRwA06szhdKvz4RTAxr8p1hpk1/ESENF2dVnStjR5+OCGTOXHKY/ArX425UYXBKRECLjw9j"+
6219 "z9N+5j+ppoG1Bb65rsHTRHJDL5pFBFTmq9/dB+pMhzlyPNPCyDxiSRK53tsB5011xJMj5EC"+
6220 "I+ykpBJJGDjPq4BDYpXnhKceAR+27MqxZEC/gAnwLzW8D71VCVJ3GyR91NcMC/j8E9AjE0"+
6221 "m5yZs+osGcmLE0NB6zK7BRQYSPoDEhaz3Jw54C9P89jyK06mwt.eahzm4Ndy35Ha3yVA"+
6222 "ZWW5TRKpEpm4Ee2zGkngYj0zktTJ08sw55jVZXNTSsFQHYM88GuvQfhe86GLaUj0D"+
6223 "KY61+Ye0ZyYui1079ctiCiH51c/iLxhBY9+0waGLhOUX280DdehR6BhNYD07Hr5gS3S1"+
6224 "Pd3W0GCHWd48D1xPpYQeQMSBuQmTeyLTi9N37LC7f1DTunY1UzB+vwIFBa5/gzp8DgpQ"+
6225 "CjJseSHPAr/eHvWJ3/tv3Xb692pwC8J9MANw0e9x7fANZIV6dR3f3U+aeVCBy0pUHYg4Dj"+
6226 "yUfBj0UacOASX5Zwz1/5ksu2JuxJyA7i18y4Vc7syVkaqctQAVOXI/wJxj3G5Yd0i+G"+
6227 "2+z5nvg1LsrQMKMrMrzGicCfPh9jbsjagXC+bD7xn2pLc78MqOCrcuIDP9Kwn941ex8R"+
6228 "gtdU2s8VYsYeI/pcBrfkFNTBgnP9RdRPXujqhn0VevcYoKpW05vgiv8Khz9/k7RDKUoxrZ"+
6229 "Qm58uA0uzxLRFPwG0yxB/g9iVb59RLM+PQ7oX2M8A1rb6Ddxu/sMeb/ogrCO3nANKs5b"+
6230 "DxfHgLA0N0Qsa0pxn9K3VfGud0XMS40E2La+2j9W06SIRvOXFnsPgoT821nAKVh3S038C"+
6231 "xvKNOCGoppBknVehLH4quTzWdK9T1gsrL4110xj1eLXbZ3ums81uecPmPCE1ahdeeVws"+
6232 "z71+zfa2UvuhWbAeA0a0N0mHlmd8dDpXue21QU/Quyyft+9U6T236BP+kC7ARqC6NqdeE"+
6233 "mNXQKUMt7WRPwt3IT1L23z2f3hVCKFSHBCN6+/qjD0qz6UeT9VEofa7vq7m5tI/dD"+
6234 "295Gh25YkHSDyq9t4XkHq03sYf/3Eha+AxrWnBhe6u6qXUUBAKSdC0VK4G7eJOYsm2H"+
6235 "REMNKEP9Wdh18hly/wA+rIDRjHOJZ21j+pl0A5DH40yaQkQocuvZiB14JrWHH8CD+7TA"+
6236 "0FuxPWSASOcmg4sz2Zhw5rXGyex1wny7+E/+IqFEUK/R89knnGSwJ7wAAAABJRUS5EKrJg"+
6237 "gg="";
6238
6239 GShellFavicon="data:image/png;base64,\
6240 iVBORw0KGgoAAAANSUHEUgAAAKwAAAB/CAYAABymylZAAAAAXNSR0IARs4c6QAAAHwLElm\
6241 TU0AKgAAAGABAAEAAUAAABAAAPgEbaAAUAAABAAARgEaOAAABAAABAAIAADpAAQAAAB\
6242 AAAATgAAAAAABIAAAQAQAAAEgAAABAAOQAQADAAAAQAABAAQAgAEAAAAQAQAAyAgW\
6243 AAAAQAAAH8AAAAAct6tZwAAAA1wSFLzAALeWAAACMBAAJcGAADQRJREFUeAhtnO9wFNU\
6244 x9/b2l+zYCKCIK1AmW1j/hj6BCKstFEFthlIGpRwdstQoqEuntR2nFqOlyTlIatInzO\
6245 amdAgY6jIyOXi7kglarVv74b3BAQpKbVajJ3e3r94WcJpe93cmcbj784kd/v7723+3nF\
6246 ffv+nx88IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6247 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6248 XiEut65tAHEwaD8Imp2wTLTadyBmzrT+42pzRSrd3peQvXpMtr.hgNcN8fewHFXUap+zyH\
6249 UASIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6250 hsF7VveE6PyDz+oM6JAmXwzZn6REVECG54SQXwhiL8tXtFfcuWqHx7AMRgIKQAIKQAJHIA\
6251 nbqem3098iHOOS8sa5044Umz+EzceAE/FWHXfnj5oFLd/Y802NJKRALvAWlqEGG4/BS\
6252 rgK0AMB/d3uCl1WJsIyVnsIyOKA08FvEYmSjYR5F3cmqUmL6v/fDd1QAV487EBP2YQ\
6253 65p9Scod24QncHyz1L1ZeDt1K7m51Ayw2yWmXNDa8cLcpj1Ww01o1U1/s305692nzqB\
6254 ZDBH2zDxP+28K1eYc9j9FveEhG5QVCIn0EvQgFL1Ka7gpKwBUnKfP0oos414tKyWAGP\
6255 e0LCe29t+zaIp1162r66Sfapw8GmV08F8fupDhKz1eSbawFudhZy1eSbawFudhZy1eS\
6256 FawE19Q0t4Cef7yOroBV9FC6E/ELf4dDgHlaosCU4r3pZsgGLGJ5621QgU9tR\
6257 rHdXjvvnSHymysWPTg6UEzEEGJes2Wmmpj4sk38VQAt/zsLLuz5aemLHzIRvKdhpAVH\
6258 F6R5Zazf8510gmVort1sExG/oTLB9s+r5h8u0ihusYfz191fGlP2cNSytlIA2//wU0\
6259 IX87zhymPTk3oc1bXxa/DKX3bYpoeHh686jgcSECUgXALY+CVN0S08Mmni9BU0OH+oIh\
6260 zFN7pHqGbb3CkOoJ1FOz2r7GvN3d1QNRtq4570TS1hkYSjSuok6478h1pHr0614D4mU7N\
6261 4tZxw1BPtULBE6uOGw2+T9jPfnK7WUbrmu5WgppjG7K13801LuApeIwCLADauxqRoolYs4\
6262 1B7b03kUETQ4panzx0+9yONN9ANsdFQmN4oxSnokzGtn+f0CANUSnWm3unjXGA0vhZsUc6+\
6263 CGjPzDufdWGrf0n8BzEJxDyScHa+vntqfdT10QH1kcVceJ9j5VJebVqETOSfv1/ERXV9fE\
6264 74raV8+EyC/v6Wf7Xamno5Kgn60Ylem/+GqTOA6dKXNTz8weCCamh8Moubur8I5B6l\
6265 Rbhvm2Bm7h195Jv118SPpWYen9sXhL6JNe71K1+UwQWk2HcmG5M6VJZwLU1Y1Tt9TuxE\
6266 sBongishohovT/2FW5q6wWhWvNYt4r/QkuK2WP20gixhfsNyyqqYae25btBav0/Pdh\
6267 1fhpLH6fTfbbp7VC6fKUFvWmH0Z5jzct1K3TjwR1jceR0tSLrJXChUo9Bn1xfgVgk\
6268 5WzuzRZKXNz11qVU8pLmXuoBCaZpaVgvJdeL20+SUpox3S11+3idYu2NxmCUDUvAJ2K\
6269 GPe4duUar7/74/WPD77y6+A3c574A/FyENPby9Yb/cvZPn3ofrtv3PCjPGJ0FAKqet3\
6270 T7w66jEapKhY0eaEKudXFG7PWKKh72Wx453a+vBksbWt1a70hprYM9SyotQ1dvtfd2/1\
6271 Tr/7DA8W5JK8WHVTLKfuMts4CsRv411A+Y8t/zd1J0n0wIvCte7WHDH3+TazxSTJKI2K6\
6272 C8V1gcGfhK8RNEUxNc9URv0K390MEfZ7UyZa8pza05RfneJKA0/luo0hw+co5y264nCLM\
6273 TyPh01R+3pt162r66Sfapw8GmV08F8fupDhKz1eSbawFudhZy1eSbawFudhZy1eS\
6274 eQd16tyHZZ6MCT21h9JeY9+0/wj2Ar30rHPFPKytZ0q1pysvX70e62obWlyJ0ePfn3\
6275 csNywSPi3RktKHi7ky7cTCEaorst0dzvHgMI/09rVtAlhulzsy0kf9UCZDBLz1LhQgT\
6276 2yDwtDLsVHH9fDrt1h1f0gMKMF/29W2qY7k7zDuuLbutncUdLMkykR600L450y2r9\
6277 213vcxGbegYZDF3bhGAT7f3yc0VArNdIoMx/886d1mVSB1ZPT25SDnaCmHwPmfaR0Mmbf\
6278 vhdsgJNGjXhR80Qu841F/WeBP4PAL2G1gtD12u7VWBWR9p9q/ubABEYVYKl/ulF8EXGsVc\
6279 V9eGEnDQ/DSrW0YSRxiQB34Eox/ssYpD73Wk9owbTpeZP++jftSoqyoAzC6xR/ofj5QrDY\
6280 BnasW4Zhsv+2rDY95qZQAvdp5We1t/GQSUM2Yw6HgmqEPJRo/ya4au+7Gffyl+LS0KWCc+3\
6281 AjzxxVwCLD+BYJ07RA6IHTuc8jclEpNNZ5g24P8SxK09H9p552S3TmJng8uzUPTN/PC\
6282 dc2P4UFAhmsfn47H6RP12VnwzrZ5LuFLwSLBs0YF72KosQJyIz2fL0oAGkG4U6b8+BxYQ\
6283 TkKvWenYqepTgZ2ftH6qhg26/jB8aPknob59jZLh9L+084E59USUQhki65Vw6P3njyDW\
6284 85ziR5001+qAbRR6Ts0+rzbMqxv2xR0csSSmQI/fCFY7LPdZ21Jz5K+CS5dELH6ixYTWL\
6285 V1/nm4/cmbCw+nXmWee48EznelWaoF+EkyUroI1DOGpL3PawpZRGFP1nhtCOXYQ5CLPQW\
6286 RGJ4fb1+lEuy3Vnc8b2F4KV1szezFVNV56gjsQv++Yot29BUzqR/jwZD1oBJWxzE18vIX\
6287 KEFL9fdsBxp/2Xs6sgKXK3dClatfd8adBN1u0UNh1Afwg6Bw93sevgj1HMLPw/bl4a6npg\
6288 pKswLw06fTm++3M1U6WmfbD3KwYwTXWj2z0cu04j5J+6WUyJBTllp6v1okE327S/NwX\
6289 Hg+21W6Vw1r14Tvl/buNDxms71u9baqa20H5DUK1z9R9GEvdrDH351k3mz394Vgd5Yp\
6290 eBmk1/kobhYBte61/0vz/ZszsaeFLR+TnORCXY90Ka7qjBbt06ctbuV/otYiPh8saeZAR7\
6291 o1Ma0u30qYVWHWwCt1a17Nd13kXo2P7z2p70cmEPEycw8L406770b+vht2PnED+nNpY/W2\
6292 LRARTh75EJ/1V5qgfllw78tTREMD4gaU5rQ376aRSqdmx7Z+GB47ui29U0W9XJ4AnJ71\
6293 16Y+xi8sfmYcRQxul4K1ysH6Be9110YHs791/4cxhvgH2jWB1J1XxXecYQuZ0x5WDLuq\
6294 Y6xMfG2XRcb6YrTjP5N07ZuP3v9irmN4F5eSbKOH0tAb6astQ07/beUGSubPmhrC27B1\
6295 0Yqhs10JvclYo4fxKoz+kq+oajVD8sgVEr9PehP+rUKXmNKLm6VpQnUkIXzm+0PveQw\
6296 h4F8U1j9Ww0rt+64stf500Wz2dG5tcd/FZS/VXH3nagrXU1+4B2j8m8Ss/FmI9D3McBj\
6297 kRn3KXzuzgppzKZU1cbOcrjMPhQ1aBxM1gsduI315d3JLR9y0V9Np1kTIE/CQYIdW2VZ\
6298 8/KpaxKkvc1bdveIDDT0Usc+RmxmDipnxmLw78tYm6HGDGct2fgznJ+8xzuSRBvQ4zrEw9\
6299 H92685VJSHFzRdI7AAPQwzK1L7SpLurBj0QFxybyb/8dmn2//ll/qgnago2AwqF/38+WE1\
6300 14af505EXEMARKaoI2CCP2xvJNV+LMZ78LkH3V2LWvt2n9w4+6JqdkXJLqg7b1TadkwlP\
6301 nHs+QSt+H1Ew5RPUvengV20d6Nf7K0t1oF1dJ/ikUsatCEBBEIAEBEIAEBEIAEBEIAEB\
6302 EIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEB\
6303 EIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEB\
6304 EIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEB\
6305 EIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEBEIAEB\
6306
6307 ITsmoreQR="data:image/png;base64,\
6308 iVBORw0KGgoAAAANSUHEUgAAAG8AAAB/AQMAADYCWwJAAAAB1MVEVX///9BaeFhgDaAAAB\
6309 hklEwQV4jdTsa2EMAwGCMX7ICKvYjXvACBe7CArASXad1lAWgS4Hw5sEVS+MwS+ZgQ\
6310 8gc4b4DhYzww8zMSaUBHNM+Kad4QC8LDpDn0g874UpPgCizj81GF3xeLwPwHknYWeceV\
6311 UEBDxB0X2AnJueYD0zNk1QassPCKjoc4nN3E1SfWqYk6jU/vAKPhg0ALSPhe8Jr0dkDMwR\
6312 YMGSSuPyWHAR19k0tkV2sb3sdw2UCQW884Rp1A9s11Pv9cTp1NRD4Xfkin8XaOCtW6Lzq\
6313 208dW/4U2Gzq1S8gbqVmkfR1N6YXK80qLD00mLGTmVzPERA8AL9vbb01fpSOL335EYtrL\
6314 S9wiqDznhUI38v5n783/gBuUs2eLg1c8gAAAAABJRUS5ErkJggg="";
6315
6316 </script>
6317
6318 <div id="GJFactory_1" class="xxxGJFactory" style=""></div>
6319 <!--
6320 https://developer.mozilla.org/en-US/docs/Web/CSS/line-height
6321 -->
6322 <style>
6323 .GJFactory{
6324     resize:both; overflow:scroll;

```

```

6325     position:static;
6326     border:1.2px dashed #282; xborder-radius:2px;
6327     margin:0px; padding:10px !important;
6328     width:340px; height:340px;
6329     flex-wrap: wrap;
6330     color:#fff; background-color:rgba(0,0,0,0.0);
6331     line-height:0.0;
6332     xxxcolor:#22a !important;
6333     text-shadow:2px 2px #ddf;
6334 }
6335 .GJFactory h1,h2,h3,h4 {
6336     xxxcolor:#22a !important;
6337 }
6338 xxxinput {
6339     border:1px dashed #0f0; border-radius:0px;
6340 }
6341 .GJWin:hover{
6342     color:#df8 !important;
6343     background-color:rgba(32,32,160,0.8) !important;
6344     line-height:0.0;
6345 }
6346 .GJWin:active{
6347     color:#df8 !important;
6348     background-color:rgba(224,32,32,0.8) !important;
6349     line-height:0.0;
6350 }
6351 .GJWin:focus{
6352     color:#df8 !important;
6353     background-color:rgba(32,32,32,1.0) !important;
6354     line-height:0.0;
6355 }
6356 .GJWin{
6357     z-index:10000;
6358     display:inline;
6359     position:relative;
6360     flex-wrap: wrap;
6361     top:0; left:0px;
6362     width:285px !important; height:205px !important;
6363     border:1px solid #eee; border-radius:2px;
6364     margin:0px; padding:0px;
6365     font-size:8pt;
6366     line-height:0.0;
6367     color:#fff; background-color:rgba(0,0,64,0.1) !important;
6368 }
6369 .GJTab{
6370     display:inline;
6371     position:relative;
6372     top:0px; left:0px;
6373     margin:0px; padding:2px;
6374     border:0px solid #000; border-radius:2px;
6375     width:90px; height:20px;
6376     font-family:Georgia;
6377     font-size:9pt;
6378     line-height:1.0;
6379     white-space:nowrap;
6380     color:#fff; background-color:rgba(0,0,64,0.7);
6381     text-align:center;
6382     vertical-align:middle;
6383 }
6384 .GJStat:focus{
6385     color:#df8 !important;
6386     background-color:rgba(32,32,32,1.0) !important;
6387     line-height:1.0;
6388 }
6389 .GJStat{
6390     display:inline;
6391     position:relative;
6392     top:0px; left:0px;
6393     margin:0px; padding:2px;
6394     border:0px solid #00f; border-radius:2px;
6395     width:166px; height:20px;
6396     font-family:monospace;
6397     font-size:9pt;
6398     line-height:1.0;
6399     color:#fff; background-color:rgba(0,0,64,0.2);
6400     text-align:center;
6401     vertical-align:middle;
6402 }
6403 .GJIcon{
6404     display:inline;
6405     position:relative;
6406     top:0px; left:1px;
6407     border:2px solid #44a;
6408     margin:0px; padding:1px;
6409     width:13.2; height:13.2px;
6410     border-radius:2px;
6411     font-family:Georgia;
6412     font-size:13.2px;
6413     line-height:1.0;
6414     white-space:nowrap;
6415     color:#fff; background-color:rgba(32,32,160,0.8);
6416     text-align:center;
6417     vertical-align:middle;
6418     text-shadow:0px 0px;
6419 }
6420 .GJText:focus{
6421     color:#fff !important;
6422     background-color:rgba(32,32,160,0.8) !important;
6423     line-height:1.0;
6424 }
6425 .GJText{
6426     display:inline;
6427     position:relative;
6428     top:0px; left:0px;
6429     border:0px solid #000; margin:0px; padding:0px;
6430     width:280px; height:160px;
6431     border:0px;
6432     font-family:Courier New,monospace !important;
6433     font-size:8pt;
6434     line-height:1.0;
6435     white-space:pre;
6436     color:#fff; xbackground-color:rgba(0,0,64,0.5);
6437     background-color:rgba(32,32,128,0.8) !important;
6438 }
6439 .GJMode{
6440     display:inline;
6441     position:relative;
6442     top:0px; left:0px;
6443     border:0px solid #000; border-radius:0px;
6444     margin:0px; padding:0px;
6445     width:280px; height:20px;
6446     font-size:9pt;
6447     line-height:1.0;
6448     white-space:nowrap;

```

```

6449     color:#fff; background-color:rgba(0,0,64,0.7);
6450     text-align:left;
6451     vertical-align:middle;
6452 }
6453 </style>
6454
6455 <script id="gsh-script">
6456 // 2020-0909 added, permanent local storage
6457 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
6458 var MyHistory = ""
6459 Permanent = localStorage;
6460 MyHistory = Permanent.getItem('MyHistory')
6461 if( MyHistory == null ){ MyHistory = "" }
6462 d = new Date()
6463 MyHistory = d.getTime()/1000+" "+document.URL+"\n" + MyHistory
6464 Permanent.setItem('MyHistory',MyHistory)
6465 //Permanent.setItem('MyWindow',window)
6466
6467 var GJLog_Win = null
6468 var GJLog_Tab = null
6469 var GJLog_Stat = null
6470 var GJLog_Text = null
6471 var GJWin_Mode = null
6472 var FProductInterval = 0
6473
6474 var GJ_FactoryID = -1
6475 var GJFactory = null
6476 if( e = document.getElementById('GJFactory_0') ){
6477     GJFactory_1.height = 0
6478     GJFactory = e
6479     e.setAttribute('class','GJFactory')
6480     var GJ_FactoryID = 0
6481 }else{
6482     GJFactory = GJFactory_1
6483     var GJ_FactoryID = 1
6484 }
6485
6486 function GJFactory_Destroy(){
6487     gjf = GJFactory
6488     //gjf = document.getElementById('GJFactory')
6489     //alert('gjf='+gjf)
6490     if( gjf != null ){
6491         if( gjf.childNodes != null ){
6492             for( i = 0; i < gjf.childNodes.length; i++ ){
6493                 gjf.removeChild(gjf.childNodes[i])
6494             }
6495         }
6496         gjf.innerHTML = ''
6497         gjf.style.width = 0
6498         gjf.style.height = 0
6499         gjf.removeAttribute('style')
6500         GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
6501         window.clearInterval(FProductInterval)
6502         return '-- Destroy: work product destroyed'
6503     }else{
6504         return '-- Destroy: work product not exist'
6505     }
6506 }
6507
6508 var TransMode = false
6509 var OnKeyControl = false
6510 var OnKeyShift = false
6511 var OnKeyAlt = false
6512 var OnKeyJ = false
6513 var OnKeyK = false
6514 var OnKeyL = false
6515
6516 function GJWin_OnKeyUp(ev){
6517     keycode = ev.code;
6518     if( keycode == 'ShiftLeft' ){
6519         OnKeyShift = false
6520     }else
6521     if( keycode == 'ControlLeft' ){
6522         onKeyControl = false
6523     }else
6524     if( keycode == 'AltLeft' ){
6525         OnKeyAlt = false
6526     }else
6527     if( keycode == 'KeyJ' ){ OnKeyJ = false }else
6528     if( keycode == 'KeyK' ){ OnKeyK = false }else
6529     if( keycode == 'KeyL' ){ OnKeyL = false }else
6530     {
6531     }
6532     ev.preventDefault()
6533 }
6534 function and(a,b){ if(a){ if(b){ return true; } return false; } }
6535 function GJWin_OnKeyDown(ev){
6536     keycode = ev.code;
6537     mode = ''
6538     key = ''
6539     if( keycode == 'ControlLeft' ){
6540         onKeyControl = true
6541         ev.preventDefault()
6542         return;
6543     }else
6544     if( keycode == 'ShiftLeft' ){
6545         OnKeyShift = true
6546         ev.preventDefault()
6547         return;
6548     }else
6549     if( keycode == 'AltLeft' ){
6550         ev.preventDefault()
6551         OnKeyAlt = true
6552         return;
6553     }else
6554     if( keycode == 'Backquote' ){
6555         TransMode = !TransMode
6556         ev.preventDefault()
6557     }else
6558     if( and(keycode == 'Space', OnKeyShift) ){
6559         TransMode = !TransMode
6560         ev.preventDefault()
6561     }else
6562     if( keycode == 'ShiftRight' ){
6563         TransMode = !TransMode
6564     }else
6565     if( keycode == 'Escape' ){
6566         TransMode = true
6567         ev.preventDefault()
6568     }else
6569     if( keycode == 'Enter' ){
6570         TransMode = false
6571         //ev.preventDefault()
6572     }

```

```

6573 if( keycode == 'KeyJ' ){ OnKeyJ = true }else
6574 if( keycode == 'KeyK' ){ OnKeyK = true }else
6575 if( keycode == 'KeyL' ){ OnKeyL = true }else
6576 {
6577 }
6578
6579 if( ev.altKey ){ key += 'Alt+' }
6580 if( onKeyControl ){ key += 'Ctrl+' }
6581 if( onKeyShift ){ key += 'Shift+' }
6582 if( and(keycode != 'KeyJ', OnKeyJ) ){ key += 'J+' }
6583 if( and(keycode != 'KeyK', OnKeyK) ){ key += 'K+' }
6584 if( and(keycode != 'KeyL', OnKeyL) ){ key += 'L+' }
6585 key += keycode
6586
6587 if( TransMode ){
6588 //mode = "[\343\201\202r]"
6589 mode = "[\343\201\202r]"
6590 }else{
6591 mode = '[---]'
6592 }
6593 ///// /gjmode.innerHTML = "[---]"
6594 GJWin_Mode.innerHTML = mode + ' ' + key
6595 //Alert('Key'+keycode)
6596 ev.stopPropagation()
6597 //ev.preventDefault()
6598 }
6599 function GJWin_OnScroll(ev){
6600 x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
6601 y = DragStartY = gsh.getBoundingClientRect().top.toFixed(0)
6602 GJLog_append('OnScroll: x='+x+',y='+y)
6603 }
6604 document.addEventListener('scroll',GJWin_OnScroll)
6605 function GJWin_OnResize(ev){
6606 w = window.innerWidth
6607 h = window.innerHeight
6608 GJLog_append('OnResize: w='+w+',h='+h)
6609 }
6610 window.addEventListener('resize',GJWin_OnResize)
6611
6612 var DragStartX = 0
6613 var DragStartY = 0
6614 function GJWin_DragStart(ev){
6615 // maybe this is the grabbing position
6616 this.style.position = 'fixed'
6617 x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
6618 y = DragStartY = this.getBoundingClientRect().top.toFixed(0)
6619 GJLog_Stat.value = 'DragStart: x='+x+',y='+y
6620 }
6621 function GJWin_Drag(ev){
6622 x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
6623 this.style.left = x - DragStartX
6624 this.style.top = y - DragStartY
6625 this.style.zIndex = '30000'
6626 this.style.position = 'fixed'
6627 x = this.getBoundingClientRect().left.toFixed(0)
6628 y = this.getBoundingClientRect().top.toFixed(0)
6629 GJLog_Stat.value = 'x='+x+',y='+y
6630 ev.preventDefault()
6631 ev.stopPropagation()
6632 }
6633 function GJWin_DragEnd(ev){
6634 x = ev.clientX; y = ev.clientY
6635 //x = ev.pageX; y = ev.pageY
6636 this.style.left = x - DragStartX
6637 this.style.top = y - DragStartY
6638 this.style.zIndex = '30000'
6639 this.style.position = 'fixed'
6640 if( true ){
6641 console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y='+y
6642 + ' parent='+this.parentNode.id)
6643 }
6644 x = this.getBoundingClientRect().left.toFixed(0)
6645 y = this.getBoundingClientRect().top.toFixed(0)
6646 GJLog_Stat.value = 'x='+x+',y='+y
6647 ev.preventDefault()
6648 ev.stopPropagation()
6649 }
6650 function GJWin_DragIgnore(ev){
6651 ev.preventDefault()
6652 ev.stopPropagation()
6653 }
6654 // 2020-09-15 let every object have console view!
6655 var GJ_ConsoleID = 0
6656 var PrevReport = new Date()
6657 function GJLog_StatUpdate(){
6658 txa = GJLog_Stat;
6659 if( txa == null ){
6660 return;
6661 }
6662 tmlap0 = new Date();
6663 p = txa.parentNode;
6664 pw = txa.getBoundingClientRect().width;
6665 ph = txa.getBoundingClientRect().height;
6666 //txa.value += '#'+p.id+' pw='+pw+' ph='+ph+'\n';
6667 txl = '#'+p.id+' pw='+pw+' ph='+ph+'\n';
6668
6669 w = txa.getBoundingClientRect().width;
6670 h = txa.getBoundingClientRect().height;
6671 //txa.value += 'w='+w+', h='+h+'\n';
6672 txl += 'w='+w+', h='+h+'\n';
6673
6674 //txa.value += '\n';
6675 //txa.value += DateShort() + '\n';
6676 txl += '\n';
6677 txl += DateShort() + '\n';
6678 tmlap1 = new Date();
6679
6680 txa.value += txl;
6681 tmlap2 = new Date();
6682
6683 // vertical centering of the last line
6684 sHeight = txa.scrollHeight - 30; // depends on the font-size
6685 tmlap3 = new Date();
6686
6687 txa.scrollTop = sHeight; // depends on the font-size
6688 tmlap4 = new Date();
6689
6690 now = tmlap0.getTime();
6691 if( PrevReport == 0 || 10000 <= now-PrevReport ){
6692 PrevReport = now;
6693 console.log('StatBarUpdate: '
6694 + 'len=' + txa.value.length + ' byte, '
6695 + 'time=' + (tmlap4 -tmlap0) + 'ms { '
6696 + 'tadd=' + (tmlap2 -tmlap1) + ', '

```

```

6697     + 'hcal=' + (tmLap3 -tmLap2) + ', '
6698     + 'scl=' + (tmLap4 -tmLap3) + '}'
6699     );
6700 }
6701 }
6702 GJWin_StatUpdate = GJLog_StatUpdate;
6703 function GJ_showTime1(wid){
6704 //e = document.getElementById(wid);
6705 //console.log(wid.id+' .value.length='+wid.value.length)
6706 if( e != null ){
6707 //e.value = DateShort();
6708 }else{
6709 // should remove the Listener
6710 }
6711 }
6712 function GJWin_OnResizeTextarea(ev){
6713 this.value += 'resized:' + '\n'
6714 }
6715 function GJ_NewConsole(wname){
6716 wid = wname + '_' + GJ_ConsoleID
6717 GJ_ConsoleID += 1
6718 }
6719 GJFactory.style.setProperty('width',360+'px'); //GJFsize
6720 GJFactory.style.setProperty('height',320+'px')
6721 e = GJFactory;
6722 console.log('GJFa #' +e.id+ ' from w='+e.style.width+', h='+e.style.height)
6723
6724 if( GJFactory.innerHTML == "" ){
6725 GJFactory.innerHTML = '<'+'H3>GJ_Factory_' + GJ_FactoryID + '<'+'H3><'+'hr>\n'
6726 }else{
6727 GJFactory.innerHTML += '<'+'hr>\n'
6728 }
6729
6730 gjwin = GJLog_Win = document.createElement('span')
6731 gjwin.id = wid
6732 gjwin.setAttribute('class', 'GJWin')
6733 gjwin.setAttribute('draggable', 'true')
6734 gjwin.addEventListener('dragstart', GJWin_DragStart)
6735 gjwin.addEventListener('drag', GJWin_Drag)
6736 gjwin.addEventListener('dragend', GJWin_Drag)
6737 gjwin.addEventListener('dragover', GJWin_DragIgnore)
6738 gjwin.addEventListener('dragenter', GJWin_DragIgnore)
6739 gjwin.addEventListener('dragleave', GJWin_DragIgnore)
6740 gjwin.addEventListener('dragexit', GJWin_DragIgnore)
6741 gjwin.addEventListener('drop', GJWin_DragIgnore)
6742 gjwin.addEventListener('keydown', GJWin_OnKeyDown)
6743
6744 gjtab = GJLog_Tab = document.createElement('textarea')
6745 gjtab.addEventListener('keydown', GJWin_OnKeyDown)
6746 gjtab.style.readonly = true
6747 gjtab.contenteditable = false
6748 gjtab.value = wid
6749 gjtab.id = wid + '_Tab'
6750 gjtab.setAttribute('class', 'GJTab')
6751 gjtab.setAttribute('spellcheck', 'false')
6752 gjwin.appendChild(gjtab)
6753
6754 gjstat = GJLog_Stat = document.createElement('textarea')
6755 gjstat.addEventListener('keydown', GJWin_OnKeyDown)
6756 gjstat.id = wid + '_Stat'
6757 gjstat.value = DateShort()
6758 gjstat.setAttribute('class', 'GJStat')
6759 gjstat.setAttribute('spellcheck', 'false')
6760 gjwin.appendChild(gjstat)
6761
6762 gjicon = document.createElement('span')
6763 gjicon.addEventListener('keydown', GJWin_OnKeyDown)
6764 gjicon.id = wid + '_Icon'
6765 gjicon.innerHTML = '<G<font color="#E44">J</font>'
6766 gjicon.setAttribute('class', 'GJIcon')
6767 gjicon.setAttribute('spellcheck', 'false')
6768 gjwin.appendChild(gjicon)
6769
6770 gjtext = GJLog_Text = document.createElement('textarea')
6771 gjtext.addEventListener('keydown', GJWin_OnKeyDown)
6772 gjtext.addEventListener('keyup', GJWin_OnKeyUp)
6773 gjtext.addEventListener('resize', GJWin_OnResizeTextarea)
6774 gjtext.id = wid + '_Text'
6775 gjtext.setAttribute('class', 'GJText')
6776 gjtext.setAttribute('spellcheck', 'false')
6777 gjwin.appendChild(gjtext)
6778
6779
6780 // user's mode as of IME
6781 gjmode = GJWin_Mode = document.createElement('textarea')
6782 gjmode.addEventListener('keydown', GJWin_OnKeyDown)
6783 gjmode.addEventListener('keydown', GJWin_OnKeyDown)
6784 gjmode.id = wid + '_Mode'
6785 gjmode.setAttribute('class', 'GJMode')
6786 gjmode.setAttribute('spellcheck', 'false')
6787 gjmode.innerHTML = '[---]'
6788 gjwin.appendChild(gjmode)
6789
6790 gjwin.zIndex = 30000
6791 GJFactory.appendChild(gjwin)
6792
6793 gjtab.scrollTop = 0
6794 gjstat.scrollTop = 0
6795
6796 //x = gjwin.getBoundingClientRect().left.toFixed(0)
6797 //y = gjwin.getBoundingClientRect().top.toFixed(0)
6798 //gjwin.style.position = 'static'
6799 //gjwin.style.left = 0
6800 //gjwin.style.top = 0
6801
6802 //update = '{'+wid+'.value=DateShort()}',
6803 update = '{GJ_showTime1('+wid+')}'
6804 // 2020-09-19 this causes memory leaks
6805 //FProductInterval = window.setInterval(update,200)
6806 //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
6807 //FProductInterval = window.setInterval(GJ_showTime1,200,wid);
6808 FProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
6809 return update
6810 }
6811 function xxxGJF_StripeClass(){
6812 GJLog_Win.style.removeProperty('width')
6813 GJLog_Tab.style.removeProperty('width')
6814 GJLog_Stat.style.removeProperty('width')
6815 GJLog_Text.style.removeProperty('width')
6816 return "Stripped classes"
6817 }
6818 function isElem(id){
6819 return document.getElementById(id) != null
6820 }

```

```

6821 function GJLog_append(...args){
6822     txt = GJLog_Text;
6823     if( txt == null ){
6824         return; // maybe GJLog element is removed
6825     }
6826     logs = args.join(' ');
6827     txt.value += logs + '\n'
6828     txt.scrollTop = txt.scrollHeight
6829     //GJLog_Stat.value = DateShort()
6830 }
6831 //window.addEventListener('time',GJLog_StatUpdate)
6832 function test_GJ_Console(){
6833     window.setInterval(GJLog_StatUpdate,1000);
6834     GJ_NewConsole('GJ_Console')
6835     e = GJFactory;
6836     console.log('GJF0 #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
6837     e.style.width = 360; //GJFsize
6838     e.style.height = 320;
6839     console.log('GJF0 #' + e.id + ' to w=' + e.style.width + ', h=' + e.style.height)
6840 }
6841 // test_GJ_Console();
6842
6843 var StopConsoleLog = true
6844 // 2020-09-15 added,
6845 // log should be saved to permanet memory
6846 // const px = new Proxy(console.log,{ alert() })
6847 __console_log = console.log
6848 __console_info = console.info
6849 __console_warn = console.warn
6850 __console_error = console.error
6851 __console_exception = console.exception
6852 // should pop callstack info.
6853 console.exception = function(...args){
6854     __console_exception(...args)
6855     alert('-- got console.exception("' + args + "' )')
6856 }
6857 console.error = function(...args){
6858     __console_error(...args)
6859     alert('-- got console.error("' + args + "' )')
6860 }
6861 console.warn = function(...args){
6862     __console_warn(...args)
6863     alert('-- got console.warn("' + args + "' )')
6864 }
6865 console.info = function(...args){
6866     alert('-- got console.info("' + args + "' )')
6867     __console_info(...args)
6868 }
6869 console.log = function(...args){
6870     __console_log(...args)
6871     if( StopConsoleLog ){
6872         return;
6873     }
6874     if( 0 <= args[0].indexOf('!') ){
6875         //alert('-- got console.log("' + args + "' )')
6876     }
6877     GJLog_append(...args)
6878 }
6879
6880 //document.getElementById('GshFaviconURL').href = GShellFavicon
6881 document.getElementById('GshFaviconURL').href = GShellInsideIcon
6882 //document.getElementById('GshFaviconURL').href = ITSmoreQR
6883 //document.getElementById('GshFaviconURL').href = GSellLogo
6884
6885 // id of GShell HTML elemets
6886 var E_BANNER = "GshBanner" // banner element in HTML
6887 var E_FOOTER = "GshFooter" // footer element in HTML
6888 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
6889 var E_GOCODE = "gsh-gocode" // Golang code of GShell
6890 var E_TODO = "gsh-todo" // TODO of GShell
6891 var E_DICT = "gsh-dict" // Dictionary of GShell
6892
6893 function bannerElem(){ return document.getElementById(E_BANNER); }
6894 function bannerStyleFunc(){ return bannerElem().style; }
6895 var bannerStyle = bannerStyleFunc()
6896 bannerStyle.backgroundImage = "url("+GSellLogo+")";
6897 //bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
6898 //bannerStyle.backgroundImage = "url("+GShellFavicon+")";
6899 GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
6900
6901 function footerElem(){ return document.getElementById(E_FOOTER); }
6902 function footerStyle(){ return footerElem().style; }
6903 //footerElem().style.backgroundImage="url("+ITSmoreQR+")";
6904 //footerStyle().backgroundImage = "url("+ITSmoreQR+")";
6905
6906 function html_fold(e){
6907     if( e.innerHTML == "Fold" ){
6908         e.innerHTML = "UnFold"
6909         document.getElementById('gsh-menu-exit').innerHTML=""
6910         document.getElementById('GshStatement').open=false
6911         GshFeatures.open = false
6912         document.getElementById('html-src').open=false
6913         document.getElementById(E_GINDEX).open=false
6914         document.getElementById(E_GOCODE).open=false
6915         document.getElementById(E_TODO).open=false
6916         document.getElementById('References').open=false
6917     }else{
6918         e.innerHTML = "Fold"
6919         document.getElementById('GshStatement').open=true
6920         GshFeatures.open = true
6921         document.getElementById(E_GINDEX).open=true
6922         document.getElementById(E_GOCODE).open=true
6923         document.getElementById(E_TODO).open=true
6924         document.getElementById('References').open=true
6925     }
6926 }
6927 function html_pure(e){
6928     if( e.innerHTML == "Pure" ){
6929         document.getElementById('gsh').style.display=true
6930         //document.style.display = false
6931         e.innerHTML = "Unpure"
6932     }else{
6933         document.getElementById('gsh').style.display=false
6934         //document.style.display = true
6935         e.innerHTML = "Pure"
6936     }
6937 }
6938
6939 var bannerIsStopping = false
6940 //NOTE: .com/JSREF/prop_style_backgroundposition.asp
6941 function shiftBG(){
6942     bannerIsStopping = !bannerIsStopping
6943     bannerStyle.backgroundPosition = "0 0";
6944 }

```



```

6945 // status should be inherited on Window Fork(), so use the status in DOM
6946 function html_stop(e,toggle){
6947     if( toggle ){
6948         if( e.innerHTML == "Stop" ){
6949             bannerIsStopping = true
6950             e.innerHTML = "Start"
6951         }else{
6952             bannerIsStopping = false
6953             e.innerHTML = "Stop"
6954         }
6955     }else{
6956         // update JavaScript variable from DOM status
6957         if( e.innerHTML == "Stop" ){ // shown if it's running
6958             bannerIsStopping = false
6959         }else{
6960             bannerIsStopping = true
6961         }
6962     }
6963 }
6964 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
6965 //html_stop(bannerElem(),false) // onInit.
6966
6967 //https://www.w3schools.com/jsref/met_win_setinterval.asp
6968 function shiftBanner(){
6969     var now = new Date().getTime();
6970     //console.log("now="+now%10)
6971     if( !bannerIsStopping ){
6972         bannerStyle.backgroundPosition = ((now/10)%100000)+" 0";
6973     }
6974 }
6975 window.setInterval(shiftBanner,10); // onInit.
6976
6977 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
6978 // from embedded html to standalone page
6979 var MyChildren = 0
6980 function html_fork(){
6981     GJFactory_Destroy()
6982     MyChildren += 1
6983     WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
6984     newwin = window.open("",WinId,"");
6985     src = document.getElementById("gsh");
6986     srchtml = src.outerHTML
6987     newwin.document.write("/*<"+"html>\n");
6988     newwin.document.write(srchtml);
6989     newwin.document.write("<"+"html>\n");
6990     newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
6991     newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
6992     newwin.document.close();
6993     newwin.focus();
6994 }
6995 function html_close(){
6996     window.close()
6997 }
6998 function win_jump(win){
6999     //win = window.top;
7000     win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
7001     if( win == null ){
7002         console.log("jump to window.opener("+win+") (Error)\n")
7003     }else{
7004         console.log("jump to window.opener("+win+")\n")
7005         win.focus();
7006     }
7007 }
7008
7009 // 0.2.9 2020-0902 created chekosum of HTML
7010 CRC32UNIX = 0x04C11DB7 // Unix cksum
7011 function byteCRC32add(bigcrc,octstr,octlen){
7012     var crc = new Uint32Array(1)
7013     crc[0] = bigcrc
7014
7015     let oi = 0
7016     for( ; oi < octlen; oi++){
7017         var oct = new Uint8Array(1)
7018         oct[0] = octstr[oi]
7019         for( bi = 0; bi < 8; bi++){
7020             //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+":"+bi+"]\n")
7021             ovf1 = crc[0] < 0 ? 1 : 0
7022             ovf2 = oct[0] < 0 ? 1 : 0
7023             ovf = ovf1 ^ ovf2
7024             oct[0] <= 1
7025             crc[0] <= 1
7026             if( ovf ){ crc[0] ^= CRC32UNIX }
7027         }
7028     }
7029     //console.log("--CRC32 byteAdd return crc="+crc[0]+","+oi+"/"+"octlen+"\n")
7030     return crc[0];
7031 }
7032 function strCRC32add(bigcrc,stri,strlen){
7033     var crc = new Uint32Array(1)
7034     crc[0] = bigcrc
7035     var code = new Uint8Array(strlen);
7036     for( i = 0; i < strlen; i++){
7037         code[i] = stri.charCodeAtAt(i) // not charAt() !!!!
7038         //console.log("=== "+code[i].toString(16)+" <== "+stri[i+"\n")
7039     }
7040     crc[0] = byteCRC32add(crc,code,strlen)
7041     //console.log("--CRC32 strAdd return crc="+crc[0]+\n")
7042     return crc[0]
7043 }
7044 function byteCRC32end(bigcrc,len){
7045     var crc = new Uint32Array(1)
7046     crc[0] = bigcrc
7047     var slen = new Uint8Array(4)
7048     let li = 0
7049     for( ; li < 4; ){
7050         slen[li] = len
7051         li += 1
7052         len >= 8
7053         if( len == 0 ){
7054             break
7055         }
7056     }
7057     crc[0] = byteCRC32add(crc[0],slen,li)
7058     crc[0] ^= 0xFFFFFFFF
7059     return crc[0]
7060 }
7061 function strCRC32(stri,len){
7062     var crc = new Uint32Array(1)
7063     crc[0] = 0
7064     crc[0] = strCRC32add(0,stri,len)
7065     crc[0] = byteCRC32end(crc[0],len)
7066     //console.log("--CRC32 "+crc[0]+" "+len+"\n")
7067     return crc[0]
7068 }

```

```

7069
7070 DestroyGJLink = null; // to be replaced
7071 DestroyFooter = null; // to be defined
7072 DestroyEventSharingCodeview = function dummy(){}
7073 Destroy_WirtualDesktop = function(){}
7074 DestroyNaviButtons = function(){}
7075
7076 function getSourceText(){
7077     if( DestroyFooter != null ) DestroyFooter();
7078     version = document.getElementById('GshVersion').innerHTML
7079     sfavico = document.getElementById('GshFaviconURL').href;
7080     sbanner = document.getElementById('GshBanner').style.backgroundImage;
7081     spositi = document.getElementById('GshBanner').style.backgroundPosition;
7082
7083     if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
7084     if( DestroyGJLink != null ) DestroyGJLink();
7085     DestroyEventSharingCodeview();
7086     Destroy_WirtualDesktop();
7087     DestroyIndexBar();
7088     DestroyNaviButtons();
7089
7090     // these should be removed by CSS selector or class, after seavaed to non-printed attribute
7091     GshBanner.removeAttribute('style');
7092     document.getElementById('GshMenuSign').removeAttribute("style");
7093     styleGMenu = GMenu.getAttribute("style")
7094     GMenu.removeAttribute("style");
7095     styleGStat = GStat.getAttribute("style")
7096     GStat.removeAttribute("style");
7097     styleGTop = GTop.getAttribute("style")
7098     GTop.removeAttribute("style");
7099     styleGshGrid = GshGrid.getAttribute("style")
7100     GshGrid.removeAttribute("style");
7101     //styleGPos = GPos.getAttribute("style");
7102     //GPos.removeAttribute("style");
7103     //GPos.innerHTML = "";
7104     //styleGLog = GLog.getAttribute("style");
7105     //GLog.removeAttribute("style");
7106     //GLog.innerHTML = "";
7107     styleGShellPlane = GShellPlane.getAttribute("style")
7108     GShellPlane.removeAttribute("style")
7109     styleRawTextViewer = RawTextViewer.getAttribute("style")
7110     RawTextViewer.removeAttribute("style")
7111     styleRawTextViewerClose = RawTextViewerClose.getAttribute("style")
7112     RawTextViewerClose.removeAttribute("style")
7113
7114     GshFaviconURL.href = "";
7115
7116     //it seems that interHTML and outerHTML generate style="" for these (??)
7117     //GshBanner.removeAttribute('style');
7118     //GshFooter.removeAttribute('style');
7119     //GshMenuSign.removeAttribute('style');
7120     GshBanner.style=""
7121     GshMenuSign.style=""
7122
7123     textarea = document.createElement("textarea")
7124     srchtml = document.getElementById("gsh").outerHTML;
7125     //textarea = document.createElement("textarea")
7126     // 2020-0910 ?? ... this causes inserting style="" to Banner and Footer,
7127     // with Chromium?/ after reloading from file:///
7128     textarea.innerHTML = srchtml
7129     // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7130     var rawtext = textarea.value
7131     //textarea.destroy()
7132     //rawtext = gsh.textContent // this removes #include <FILENAME> too
7133     var orgtext = ""
7134     + "/*<" + "html>\n" // lost preamble text
7135     + rawtext
7136     + "<" + "html>\n" // lost trail text
7137     ;
7138
7139     tlen = orgtext.length
7140     //console.log("getSourceText: length="+tlen+"\n")
7141     document.getElementById('GshFaviconURL').href = sfavico;
7142
7143     document.getElementById('GshBanner').style.backgroundImage = sbanner;
7144     document.getElementById('GshBanner').style.backgroundPosition = spositi;
7145
7146     GStat.setAttribute("style",styleGStat)
7147     GMenu.setAttribute("style",styleGMenu)
7148     GTop.setAttribute("style",styleGTop)
7149     //GLog.setAttribute("style",styleGLog)
7150     //GPos.setAttribute("style",styleGPos)
7151     GshGrid.setAttribute("style",styleGshGrid)
7152     GShellPlane.setAttribute("style",styleGShellPlane)
7153     RawTextViewer.setAttribute("style",styleRawTextViewer)
7154     RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
7155     canontext = orgtext.replace(' style=""','')
7156     // open="" too
7157     return canontext
7158 }
7159 function getDigest(){
7160     var text = ""
7161     text = getSourceText()
7162     var digest = ""
7163     tlen = text.length
7164     digest = strCRC32(text,tlen) + " " + tlen
7165     return { text, digest }
7166 }
7167 function html_digest(){
7168     version = document.getElementById('GshVersion').innerHTML
7169     let {text, digest} = getDigest()
7170     alert("cksum: " + digest + " " + version)
7171 }
7172 function charsin(stri,char){
7173     ln = 0;
7174     for( i = 0; i < stri.length; i++){
7175         if( stri.charCodeAtAt(i) == char.charCodeAtAt(0) )
7176             ln++;
7177     }
7178     return ln;
7179 }
7180
7181 //class digestElement extends HTMLElement { }
7182 //< script>customElements.define('digest',digestElement)< /script>
7183 function showDigest(e){
7184     result = 'version=' + GshVersion.innerHTML + '\n'
7185     result += 'lines=' + e.dataset.lines + '\n'
7186     + 'length=' + e.dataset.length + '\n'
7187     + 'crc32u=' + e.dataset.crc32u + '\n'
7188     + 'time=' + e.dataset.time + '\n';
7189
7190     alert(result)
7191 }
7192

```

```

7193 function html_sign(e){
7194     if( RawTextViewer.style.zIndex == 1000 ){
7195         hideRawTextViewer()
7196         return
7197     }
7198     DestroyIndexBar();
7199     DestroyNaviButtons();
7200     DestroyEventSharingCodeview();
7201     Destroy_WirtualDesktop();
7202     GJFactory_Destroy()
7203     if( DestroyGJLink != null ) DestroyGJLink();
7204     //gsh_digest_.innerHTML = "";
7205     text = getSourceText() // the original text
7206     tlen = text.length
7207     digest = strCRC32(text,tlen)
7208     //gsh_digest_.innerHTML = digest + " " + tlen
7209     //text = getSourceText() // the text with its digest
7210     Lines = charsin(text,'\n')
7211
7212     name = "gsh"
7213     sid = name + "-digest"
7214     d = new Date()
7215     signedAt = d.getTime()
7216
7217     sign = '/'+'*'+span'\n'
7218     + ' id="' + sid + '"\n'
7219     + ' class="digest_"\n'
7220     + ' data-target-id="'+name+'\n'
7221     + ' data-crc32u="' + digest + '"\n'
7222     + ' data-length="' + tlen + '"\n'
7223     + ' data-lines="' + Lines + '"\n'
7224     + ' data-time="' + signedAt + '"\n'
7225     + '>' + '/span>\n'+'\n'
7226
7227     text = sign + text
7228
7229     txthtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td>'
7230     + '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">'
7231     for( i = 1; i <= Lines; i++){
7232         txthtml += i.toString() + '\n'
7233     }
7234     txthtml += ""
7235     + '<' + '/textarea>'
7236     + '<' + '/td><' + 'td>'
7237     + '<' + 'textarea cols=150 rows=' + Lines + 'spellcheck="false"'
7238     + ' class="LineNumbered">'
7239     + text + '<' + '/textarea>'
7240     + '<' + '/td><' + 'tr><' + '/table>'
7241
7242     for( i = 1; i <= 30; i++){
7243         txthtml += '<br>\n'
7244     }
7245     RawTextViewer.innerHTML = txthtml
7246     RawTextViewer.spellcheck = false // (spellcheck above seems ineffective)
7247
7248     btn = e
7249     e.style.color = "rgba(128,128,255,0.9)";
7250     y = e.getBoudingClientRect().top.toFixed(0)
7251     //h = e.getBoudingClientRect().height.toFixed(0)
7252     RawTextViewer.style.top = Number(y) + 30
7253     RawTextViewer.style.left = 100;
7254     RawTextViewer.style.height = window.innerHeight - 20;
7255     //RawTextViewer.style.Opacity = 1.0;
7256     //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
7257     RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
7258     RawTextViewer.style.zIndex = 1000;
7259     RawTextViewer.style.display = true;
7260
7261     if( RawTextViewerClose.style == null ){
7262         RawTextViewerClose.style = "";
7263     }
7264     RawTextViewerClose.style.top = Number(y) + 10
7265     RawTextViewerClose.style.left = 100;
7266     RawTextViewerClose.style.zIndex = 1001;
7267
7268     ScrollToElement(CurElement,RawTextViewerClose)
7269 }
7270 function hideRawTextViewer(){
7271     RawTextViewer.style.left = 10000;
7272     RawTextViewer.style.zIndex = -100;
7273     RawTextViewer.style.Opacity = 0.0;
7274     RawTextViewer.style = null
7275     RawTextViewer.innerHTML = "";
7276
7277     GshMenuSign.style.color = "rgba(255,128,128,1.0)";
7278     RawTextViewerClose.style.top = 0;
7279     RawTextViewerClose.style = null
7280 }
7281
7282 // source code viewr
7283 function frame_close(){
7284     srcframe = document.getElementById("src-frame");
7285     srcframe.innterHTML = "";
7286     //srcframe.style.cols = 1;
7287     srcframe.style.rows = 1;
7288     srcframe.style.height = 0;
7289     srcframe.style.display = false;
7290     src = document.getElementById("SrcTextarea");
7291     src.innerHTML = ""
7292     //src.cols = 0
7293     src.rows = 0
7294     src.display = false
7295     //alert("--closed--")
7296 }
7297 //<!-- | <span onclick="html_view();">Source</span> -->
7298 //<!-- | <span onclick="frame_close();">SourceClose</span> -->
7299 //<!-- | <span>Download</span> -->
7300 function frame_open(){
7301     DestroyIndexBar();
7302     DestroyNaviButtons();
7303     if( DestroyFooter != null ) DestroyFooter();
7304     document.getElementById('GshFaviconURL').href = "";
7305     oldsrc = document.getElementById("GENSRC");
7306     if( oldsrc != null ){
7307         //alert("--I--(erasing old text)")
7308         oldsrc.innterHTML = "";
7309         return
7310     }else{
7311         //alert("--I--(no old text)")
7312     }
7313     styleBanner = GshBanner.getAttribute("style")
7314     GshBanner.removeAttribute("style")
7315     if( document.getElementById('GJC_1') ){ GJC_1.remove() }
7316

```

```

7317 GshFaviconURL.href = "";
7318 GStat.removeAttribute('style')
7319 GshGrid.removeAttribute('style')
7320 GshMenuSign.removeAttribute('style')
7321 //GPos.removeAttribute('style')
7322 //GPos.innerHTML = "";
7323 //GLog.removeAttribute('style')
7324 //GLog.innerHTML = "";
7325 GMenu.removeAttribute('style')
7326 GTop.removeAttribute('style')
7327 GShellPlane.removeAttribute('style')
7328 RawTextViewer.removeAttribute('style')
7329 RawTextViewerClose.removeAttribute('style')
7330
7331 if( DestroyGJLink != null ) DestroyGJLink();
7332 GJFactory_Destroy()
7333 Destroy_WirtualDesktop();
7334 DestroyEventSharingCodeview();
7335
7336 src = document.getElementById("gsh");
7337 srchtml = src.outerHTML
7338 srcframe = document.getElementById("src-frame");
7339 srcframe.innerHTML = ""
7340 + "<"+<cite id="GENSRC">\n
7341 + "<"+<style>\n
7342 + "#GENSRC textarea{tab-size:4;}\n"
7343 + "#GENSRC textarea{-o-tab-size:4;}\n"
7344 + "#GENSRC textarea{-moz-tab-size:4;}\n"
7345 + "#GENSRC textarea{spellcheck:false;}\n"
7346 + "</"+<style>\n"
7347 + "<"+<textarea id="SrcTextarea" cols=100 rows=20 class="gsh-code" spellcheck="false">'
7348 + "/*"+<html>\n" // lost preamble text
7349 + srchtml
7350 + "<"+</html>\n" // lost trail text
7351 + "</"+<textarea>\n"
7352 + "</"+<cite><!-- GENSRC -->\n";
7353
7354 //srcframe.style.cols = 80;
7355 //srcframe.style.rows = 80;
7356
7357 GshBanner.setAttribute('style',styleBanner)
7358 }
7359 function fill_CSSView(){
7360 part = document.getElementById('GshStyleDef')
7361 view = document.getElementById('gsh-style-view')
7362 view.innerHTML = ""
7363 + "<"+<textarea cols=100 rows=20 class="gsh-code">'
7364 + part.innerHTML
7365 + "<"+</textarea>"
7366 }
7367 function fill_JavaScriptView(){
7368 jspart = document.getElementById('gsh-script')
7369 view = document.getElementById('gsh-script-view')
7370 view.innerHTML = ""
7371 + "<"+<textarea cols=100 rows=20 class="gsh-code">'
7372 + jspart.innerHTML
7373 + "<"+</textarea>"
7374 }
7375 function fill_DataView(){
7376 part = document.getElementById('gsh-data')
7377 view = document.getElementById('gsh-data-view')
7378 view.innerHTML = ""
7379 + "<"+<textarea cols=100 rows=20 class="gsh-code">'
7380 + part.innerHTML
7381 + "<"+</textarea>"
7382 }
7383 function jumpto_StyleView(){
7384 jsview = document.getElementById('html-src')
7385 jsview.open = true
7386 jsview = document.getElementById('gsh-style-frame')
7387 jsview.open = true
7388 fill_CSSView()
7389 }
7390 function jumpto_JavaScriptView(){
7391 jsview = document.getElementById('html-src')
7392 jsview.open = true
7393 jsview = document.getElementById('gsh-script-frame')
7394 jsview.open = true
7395 fill_JavaScriptView()
7396 }
7397 function jumpto_DataView(){
7398 jsview = document.getElementById('html-src')
7399 jsview.open = true
7400 jsview = document.getElementById('gsh-data-frame')
7401 jsview.open = true
7402 fill_DataView()
7403 }
7404 function jumpto_WholeView(){
7405 jsview = document.getElementById('html-src')
7406 jsview.open = true
7407 jsview = document.getElementById('gsh-whole-view')
7408 jsview.open = true
7409 frame_open()
7410 }
7411 function html_view(){
7412 html_stop();
7413
7414 banner = document.getElementById('GshBanner').style.backgroundImage;
7415 footer = document.getElementById('GshFooter').style.backgroundImage;
7416 document.getElementById('GshBanner').style.backgroundImage = "";
7417 document.getElementById('GshBanner').style.backgroundPosition = "";
7418 document.getElementById('GshFooter').style.backgroundImage = "";
7419
7420 //srcwin = window.open("", "CodeView2", "");
7421 srcwin = window.open("", "", "");
7422 srcwin.document.write("<span id="gsh">\n");
7423
7424 src = document.getElementById("gsh");
7425 srcwin.document.write("<"+<style>\n");
7426 srcwin.document.write("<textarea{tab-size:4;}\n");
7427 srcwin.document.write("<textarea{-o-tab-size:4;}\n");
7428 srcwin.document.write("<textarea{-moz-tab-size:4;}\n");
7429 srcwin.document.write("</style>\n");
7430 srcwin.document.write("<ch2>\n");
7431 srcwin.document.write("<"+<span onclick="window.close();>\n">Close</span> | \n");
7432 //srcwin.document.write("<"+<span onclick="html_stop();>\n">Run</span>\n");
7433 srcwin.document.write("</h2>\n");
7434 srcwin.document.write("<"+<textarea id="gsh-src-src" cols=100 rows=60>"");
7435 srcwin.document.write("/*"+<html>\n");
7436 srcwin.document.write("<"+<span id="gsh">");
7437 srcwin.document.write(src.innerHTML);
7438 srcwin.document.write("<"+</span><"+</html>\n");
7439 srcwin.document.write("<"+<textarea>\n");
7440

```

```

7441 document.getElementById('GshBanner').style.backgroundImage = banner;
7442 document.getElementById('GshFooter').style.backgroundImage = footer
7443
7444 sty = document.getElementById("GshStyleDef");
7445 srcwin.document.write("<"+sty>\n");
7446 srcwin.document.write(sty.innerHTML);
7447 srcwin.document.write("<"+sty>\n");
7448
7449 run = document.getElementById("gsh-script");
7450 srcwin.document.write("<"+script>\n");
7451 srcwin.document.write(run.innerHTML);
7452 srcwin.document.write("<"+script>\n");
7453
7454 srcwin.document.write("<"+span><"+html>\n"); // gsh span
7455 srcwin.document.close();
7456 srcwin.focus();
7457 }
7458 GSH = document.getElementById("gsh")
7459
7460 //GSH.onclick = "alert('Ouch!')"
7461 //GSH.css = "{background-color:#eef;}"
7462 //GSH.style = "background-color:#eef;"
7463 //GSH.style.display = false;
7464 //alert('Ouch0!')
7465 //GSH.style.display = true;
7466
7467 // 2020-0904 created, tentative
7468 document.addEventListener('keydown',jgshCommand);
7469 //CurElement = GshStatement
7470 CurElement = GshMenu
7471 MemElement = GshMenu
7472
7473 function nextSib(e){
7474     n = e.nextSibling;
7475     for( i = 0; i < 100; i++ ){
7476         if( n == null ){
7477             break;
7478         }
7479         if( n.nodeName == "DETAILS" ){
7480             return n;
7481         }
7482         n = n.nextSibling;
7483     }
7484     return null;
7485 }
7486 function prevSib(e){
7487     n = e.previousSibling;
7488     for( i = 0; i < 100; i++ ){
7489         if( n == null ){
7490             break;
7491         }
7492         if( n.nodeName == "DETAILS" ){
7493             return n;
7494         }
7495         n = n.previousSibling;
7496     }
7497     return null;
7498 }
7499 function setColor(e,eName,eColor){
7500     if( e.hasChildNodes() ){
7501         s = e.childNodes;
7502         if( s != null ){
7503             for( ci = 0; ci < s.length; ci++ ){
7504                 if( s[ci].nodeName == eName ){
7505                     s[ci].style.color = eColor;
7506                     //s[ci].style.backgroundColor = eColor;
7507                     break;
7508                 }
7509             }
7510         }
7511     }
7512 }
7513
7514 // https://docs.microsoft.com/en-us/previous-versions/hh781509(v=vs.85)
7515 function showCurElementPosition(ev){
7516 // if( document.getElementById("GPos") == null ){
7517 //     return;
7518 // }
7519 // if( GPos == null ){
7520 //     return;
7521 // }
7522 e = CurElement
7523 y = e.getBoundingClientRect().top.toFixed(0)
7524 x = e.getBoundingClientRect().left.toFixed(0)
7525
7526 h = ev + " "
7527 h += 'y'+y+', ' + 'x'+x+' -- '
7528 h += "w" + window.innerWidth + ", h=" + window.innerHeight + " -- "
7529 //GPos.test = h
7530 //GPos.innerHTML = h
7531 // GPos.innerHTML = h
7532 }
7533
7534 function zero2(n){
7535     if( n < 10 ){
7536         return '0' + n;
7537     }else{
7538         return n;
7539     }
7540 }
7541 function DateHourMin(){
7542     d = new Date();
7543     //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes())
7544     return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
7545 }
7546 function DateShort(){
7547     d = new Date();
7548     return d.getFullYear()
7549         + "/" + zero2(d.getMonth())
7550         + "/" + zero2(d.getDate())
7551         + " " + zero2(d.getHours())
7552         + ":" + zero2(d.getMinutes())
7553         + ":" + zero2(d.getSeconds())
7554 }
7555 function DateLong0(ms){
7556     d = new Date();
7557     d.setTime(ms);
7558     return
7559         //d.getFullYear() + "/" + d.getMonth() + "/" + d.getDate() + " "
7560         //+ d.getHours() + ":" + d.getMinutes() + ":" + d.getSeconds()
7561         DateShort()
7562         + "." + d.getMilliseconds()
7563         + " " + d.getTimezoneOffset()/60
7564         + " "

```

```

7565     + d.getTime() + "." + d.getMilliseconds()
7566 }
7567 }
7568 function DateLong(){
7569     return DateLong0(new Date());
7570 }
7571 function GShellMenu(e){
7572     //GLog.innerHTML = "Hello, World! (" + DateLong() + ")";
7573     showGShellPlane()
7574 }
7575 // placements of planes
7576 function GShellResizeX(ev){
7577     //if( document.getElementById("GMenu") != null ){
7578     GMenu.style.left = window.innerWidth - 100
7579     GMenu.style.top = window.innerHeight - 90 - 200
7580     //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)
7581 }
7582 //}
7583 GStat.style.width = window.innerWidth
7584 //if( document.getElementById("GPos") != null ){
7585 //GPos.style.width = window.innerWidth
7586 //GPos.style.top = window.innerHeight - 30; //GPos.style.height
7587 //}
7588 //if( document.getElementById("GLog") != null ){
7589 // GLog.style.width = window.innerWidth
7590 //GLog.innerHTML = ""
7591 //}
7592 //if( document.getElementById("GLog") != null ){
7593 //GLog.innerHTML = "Resize: w=" + window.innerWidth +
7594 //", h=" + window.innerHeight
7595 //}
7596 showCurElementPosition(ev)
7597 }
7598 function GShellResize(){
7599     GShellResizeX("[RESIZE]")
7600 }
7601 window.onresize = GShellResize
7602 var prevNode = null
7603 var LogMouseMoveOverElement = false;
7604 function GJSH_OnMouseMove(ev){
7605     if( LogMouseMoveOverElement == false ){
7606         return;
7607     }
7608     x = ev.clientX
7609     y = ev.clientY
7610     d = new Date()
7611     t = d.getTime() / 1000
7612     if( document.elementFromPoint ){
7613         e = document.elementFromPoint(x,y)
7614         if( e != null ){
7615             if( e == prevNode ){
7616                 }else{
7617                     console.log('Mo-'+'t+' ('+'x+', '+'y+') '
7618                         +'e.nodeType+' '+'e.tagName+'#'+e.id)
7619                     prevNode = e
7620                 }
7621             }else{
7622                 console.log(t+' ('+'x+', '+'y+') no element')
7623             }
7624         }else{
7625             console.log(t+' ('+'x+', '+'y+') no elementFromPoint')
7626         }
7627     }
7628 window.addEventListener('mousemove',GJSH_OnMouseMove);
7629 }
7630 function GJSH_OnMouseMoveScreen(ev){
7631     x = ev.screenX
7632     y = ev.screenY
7633     d = new Date()
7634     t = d.getTime() / 1000
7635     console.log(t+' ('+'x+', '+'y+') no elementFromPoint')
7636 }
7637 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
7638 }
7639 function ScrollToElement(oe,ne){
7640     ne.scrollIntoView()
7641     ny = ne.getBoundingClientRect().top.toFixed(0)
7642     nx = ne.getBoundingClientRect().left.toFixed(0)
7643     //GLog.innerHTML = "["+ny+", "+nx+"]"
7644     //window.scrollTo(0,0)
7645 }
7646 GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
7647 GshGrid.style.left = '250px';
7648 GshGrid.style.zIndex = 0
7649 if( false ){
7650     oy = oe.getBoundingClientRect().top.toFixed(0)
7651     ox = oe.getBoundingClientRect().left.toFixed(0)
7652     y = e.getBoundingClientRect().top.toFixed(0)
7653     x = e.getBoundingClientRect().left.toFixed(0)
7654     window.scrollTo(x,y)
7655     ny = e.getBoundingClientRect().top.toFixed(0)
7656     nx = e.getBoundingClientRect().left.toFixed(0)
7657     //GLog.innerHTML = "["+oy+", "+ox+"]->["+y+", "+x+"]->["+ny+", "+nx+"]"
7658 }
7659 }
7660 function showGShellPlane(){
7661     if( GShellPlane.style.zIndex == 0 ){
7662         GShellPlane.style.zIndex = 1000;
7663         GShellPlane.style.left = 30;
7664         GShellPlane.style.height = 320;
7665         GShellPlane.innerHTML = DateLong() + "<br>" +
7666             "-- History --<br>" + MyHistory;
7667     }else{
7668         GShellPlane.style.zIndex = 0;
7669         GShellPlane.style.left = 0;
7670         GShellPlane.style.height = 50;
7671         GShellPlane.innerHTML = "";
7672     }
7673 }
7674 var SuppressGJShell = false
7675 function jgshCommand(kevent){
7676     if( SuppressGJShell ){
7677         return
7678     }
7679     key = kevent
7680     keycode = key.code
7681     //GStat.style.width = window.innerWidth
7682     GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
7683 }
7684 console.log("JSGsh-Key: "+keycode+"(^-^)/")
7685 if( keycode == "Slash" ){
7686     console.log(' ('+'x+', '+'y+') '
7687         e = document.elementFromPoint(x,y)
7688         console.log(' ('+'x+', '+'y+') 'e.nodeType+' '+'e.tagName+'#'+e.id)

```

```

7689 }else
7690 if( keycode == "Digit0" ){ // fold side-bar
7691 // "Zero page"
7692 showGShellPlane();
7693 }else
7694 if( keycode == "Digit1" ){ // fold side-bar
7695 primary.style.width = "94%"
7696 secondary.style.width = "0%"
7697 secondary.style.opacity = 0
7698 GStat.innerHTML = "[Single Column View]"
7699 }else
7700 if( keycode == "Digit2" ){ // unfold side-bar
7701 primary.style.width = "58%"
7702 secondary.style.width = "36%"
7703 secondary.style.opacity = 1
7704 GStat.innerHTML = "[Double Column View]"
7705 }else
7706 if( keycode == "KeyU" ){ // fold/unfold all
7707 html_fold(GshMenuFold);
7708 location.href = "#"+CurElement.id;
7709 }else
7710 if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
7711 CurElement.open = !CurElement.open;
7712 }else
7713 if( keycode == "ArrowRight" ){ // unfold the element
7714 CurElement.open = true
7715 }else
7716 if( keycode == "ArrowLeft" ){ // unfold the element
7717 CurElement.open = false
7718 }else
7719 if( keycode == "KeyI" ){ // inspect the element
7720 e = CurElement
7721 //GLog.innerHTML =
7722 GJLog_append("Current Element: " + e + "<br>"
7723 + "name="+e.nodeName + ", "
7724 + "id="+e.id + ", "
7725 + "children="+e.childNodes.length + ", "
7726 + "parent="+e.parentNode.id + "<br>"
7727 + "text="+e.textContent)
7728 GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
7729 return
7730 }else
7731 if( keycode == "KeyM" ){ // memory the position
7732 MemElement = CurElement
7733 }else
7734 if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
7735 e = nextSib(CurElement)
7736 if( e != null ){
7737 setColor(CurElement,"SUMMARY","#fff")
7738 setColor(e,"SUMMARY","#8f8") // should be complement ?
7739 oe = CurElement
7740 CurElement = e
7741 //location.href = "#"+e.id;
7742 ScrollToElement(oe,e)
7743 }
7744 }else
7745 if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
7746 oe = CurElement
7747 e = prevSib(CurElement)
7748 if( e != null ){
7749 setColor(CurElement,"SUMMARY","#fff")
7750 setColor(e,"SUMMARY","#8f8") // should be complement ?
7751 CurElement = e
7752 //location.href = "#"+e.id;
7753 ScrollToElement(oe,e)
7754 }else{
7755 e = document.getElementById("GshBanner")
7756 if( e != null ){
7757 setColor(CurElement,"SUMMARY","#fff")
7758 CurElement = e
7759 ScrollToElement(oe,e)
7760 }else{
7761 e = document.getElementById("primary")
7762 if( e != null ){
7763 setColor(CurElement,"SUMMARY","#fff")
7764 CurElement = e
7765 ScrollToElement(oe,e)
7766 }
7767 }
7768 }
7769 }else
7770 if( keycode == "KeyR" ){
7771 location.reload()
7772 }else
7773 if( keycode == "KeyJ" ){
7774 GshGrid.style.top = '120px';
7775 GshGrid.innerHTML = '><{Down}';
7776 }else
7777 if( keycode == "KeyK" ){
7778 GshGrid.style.top = '0px';
7779 GshGrid.innerHTML = '^~^{Up}';
7780 }else
7781 if( keycode == "KeyH" ){
7782 GshGrid.style.left = '0px';
7783 GshGrid.innerHTML = '{_}{Left}';
7784 }else
7785 if( keycode == "KeyL" ){
7786 //GLog.innerHTML +=
7787 GJLog_append(
7788 "screen="+screen.width+'px'+<br>'+
7789 "window="+window.innerWidth+'px'+<br>'
7790 )
7791 GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
7792 GshGrid.innerHTML = '{@}{Right}';
7793 }else
7794 if( keycode == "KeyS" ){
7795 html_stop(GshMenuStop,true)
7796 }else
7797 if( keycode == "KeyF" ){
7798 html_fork()
7799 }else
7800 if( keycode == "KeyC" ){
7801 window.close()
7802 }else
7803 if( keycode == "KeyD" ){
7804 html_digest()
7805 }else
7806 if( keycode == "KeyV" ){
7807 e = document.getElementById('gsh-digest')
7808 if( e != null ){
7809 showDigest(e)
7810 }
7811 }
7812 }

```

```

7813 showCurElementPosition("[+key.code+" --");
7814 //if( document.getElementById("GPos") != null ){
7815 //GPos.innerHTML += "[+key.code+" --"
7816 //}
7817 //GShellResizeX("[+key.code+" --");
7818 }
7819 GShellResizeX("[INIT]");
7820
7821 DisplaySize = '-- Display: ' + 'screen='+screen.width+'px, ' + 'window='+window.innerWidth+'px';
7822
7823 let {text, digest} = getDigest()
7824 //GLog.innerHTML +=
7825 GJLog_append(
7826 '-- GShell: ' + GshVersion.innerHTML + '\n' +
7827 '-- Digest: ' + digest + '\n' +
7828 DisplaySize
7829 //+ "<br>" + "-- LastVisit:<br>" + MyHistory
7830 )
7831 GShellResizeX(null);
7832
7833 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
7834 //Convert a string into an ArrayBuffer
7835 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
7836 function str2ab(str) {
7837     const buf = new ArrayBuffer(str.length);
7838     const bufView = new Uint8Array(buf);
7839     for (let i = 0, strLen = str.length; i < strLen; i++) {
7840         bufView[i] = str.charCodeAt(i);
7841     }
7842     return buf;
7843 }
7844 function importPrivateKey(pem) {
7845     const binaryDerString = window.atob(pemContents);
7846     const binaryDer = str2ab(binaryDerString);
7847     return window.crypto.subtle.importKey(
7848         "pkcs8",
7849         binaryDer,
7850         {
7851             name: "RSA-PSS",
7852             modulusLength: 2048,
7853             publicExponent: new Uint8Array([1, 0, 1]),
7854             hash: "SHA-256",
7855         },
7856         true,
7857         ["sign"]
7858     );
7859 }
7860 //importPrivateKey(ppem)
7861
7862 //key = {}
7863 //buf = "abc"
7864 //enc = "xyzxxxxxx"; //crypto.publicEncrypt(key,buf)
7865 //b64 = btoa(enc)
7866 //dec = atob(b64)
7867 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
7868 </script>
7869 */
7870
7871 /*
7872 <!-- ----- GJConsole BEGIN { ----- -->
7873 <span id="gjc" data-title="GJConsole" data-author="satoits-more.jp">
7874 <details><summary>GJ Console</summary>
7875 <p>
7876 <span id="GJE_RootNode0"></span>
7877 </p>
7878 <style id="GJConsoleStyle">
7879 .GJConsole {
7880     z-index:1000;
7881     width:400; height:200px;
7882     margin:2px;
7883     color:#fff; background-color:#66a;
7884     font-size:12px; font-family:monospace,Courier New;
7885 }
7886 </style>
7887
7888 <script id="GJConsoleScript" class="GJConsole">
7889 var PS1 = "% "
7890 function GJC_KeyDown(keyevent){
7891     key = keyevent.code
7892     if( key == "Enter" ){
7893         GJC_Command(this)
7894         this.value += "\n" + PS1 // prompt
7895     }else
7896     if( key == "Escape"){
7897         SuppressGJShell = false
7898         GshMenu.focus() // should be previous focus
7899     }
7900 }
7901 var GJC_SessionId
7902 function GJC_SetSessionId(){
7903     var xd = new Date()
7904     GJC_SessionId = xd.getTime() / 1000
7905 }
7906 GJC_SetSessionId()
7907 function GJC_Memory(mem,args,text){
7908     argv = args.split(' ')
7909     cmd = argv[0]
7910     argv.shift()
7911     args = argv.join(' ')
7912     ret = ""
7913
7914     if( cmd == 'clear' ){
7915         Permanent.setItem(mem,'')
7916     }else
7917     if( cmd == 'read' ){
7918         ret = Permanent.getItem(mem)
7919     }else
7920     if( cmd == 'save' ){
7921         val = Permanent.getItem(mem)
7922         if( val == null ){ val = "" }
7923         d = new Date()
7924         val += d.getTime()/1000+ " +GJC_SessionId+ " +document.URL+ " +args+"\n"
7925         val += text.value
7926         Permanent.setItem(mem,val)
7927     }else
7928     if( cmd == 'write' ){
7929         val = Permanent.getItem(mem)
7930         if( val == null ){ val = "" }
7931         d = new Date()
7932         val += d.getTime()/1000+ " +GJC_SessionId+ " +document.URL+ " +args+"\n"
7933         Permanent.setItem(mem,val)
7934     }else{
7935         ret = "Commands: write | read | save | clear"
7936     }
7937 }

```



```

7937     return ret
7938 }
7939 // -- 2020-09-14 added TableEditor
7940 var GJE_CurElement = null; //GJE_RootNode
7941 GJE_NodeSaved = null
7942 GJE_TableNo = 1
7943 function GJE_StyleKeyCommand(kev){
7944     keycode = kev.code
7945     console.log('GJE-Key: '+keycode)
7946     if( keycode == 'Escape' ){
7947         GJE_SetStyle(this);
7948     }
7949     kev.stopPropagation()
7950     // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
7951 }
7952 var GJE_CommandMode = false
7953 function GJE_TableKeyCommand(kev,tab){
7954     wasCmdMode = GJE_CommandMode
7955     key = kev.code
7956     if( key == 'Escape' ){
7957         console.log("To command mode: "+tab.nodeName+"#" +tab.id)
7958         //tab.setAttribute('contenteditable', 'false')
7959         tab.style.caretColor = "blue"
7960         GJE_CommandMode = true
7961     }else
7962     if( key == "KeyA" ){
7963         tab.style.caretColor = "red"
7964         GJE_CommandMode = false
7965     }else
7966     if( key == "KeyI" ){
7967         tab.style.caretColor = "red"
7968         GJE_CommandMode = false
7969     }else
7970     if( key == "KeyO" ){
7971         tab.style.caretColor = "red"
7972         GJE_CommandMode = false
7973     }else
7974     if( key == "KeyJ" ){
7975         console.log("ROW-DOWN")
7976     }else
7977     if( key == "KeyK" ){
7978         console.log("ROW-UP")
7979     }else
7980     if( key == "Keyw" ){
7981         console.log("COL-FORW")
7982     }else
7983     if( key == "Keyb" ){
7984         console.log("COL-BACK")
7985     }
7986
7987     kev.stopPropagation()
7988     if( wasCmdMode ){
7989         kev.preventDefault()
7990     }
7991 }
7992 function GJE_DragEvent(ev,elem){
7993     x = ev.clientX
7994     y = ev.clientY
7995     console.log("Dragged: "+this.nodeName+"#" +this.id+' x='+x+' y='+y)
7996 }
7997 // https://developer.mozilla.org/en-US/docs/Web/API/Event/Event
7998 // https://www.w3.org/TR/uievents/#events-mouseevents
7999 function GJE_DropEvent(ev,elem){
8000     x = ev.clientX
8001     y = ev.clientY
8002     this.style.x = x
8003     this.style.y = y
8004     this.style.position = 'absolute' // 'fixed'
8005     this.parentNode = gsh // just for test
8006     console.log("Dropped: "+this.nodeName+"#" +this.id+' x='+x+' y='+y
8007         +' parent='+this.parentNode.id)
8008 }
8009 function GJE_SetTableStyle(ev){
8010     this.innerHTML = this.value; // sync. for external representation?
8011     if(false){
8012         stid = this.parentNode.id+this.id
8013         // and remove "_span" at the end
8014         e = document.getElementById(stid)
8015         //alert('SetTableStyle #' +e.id+'\n'+this.value)
8016         if( e != null ){
8017             e.innerHTML = this.value
8018         }else{
8019             console.log('Style Not found: '+stid)
8020         }
8021         //alert('event StopPropagation: '+ev)
8022     }
8023 }
8024 function setCSSofClass(cclass,cstyle){
8025     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
8026     rlen = ss.cssRules.length;
8027     let tabrule = null;
8028     rulex = -1
8029
8030     // should skip white space at the top of cstyle
8031     sel = cstyle.charAt(0);
8032     selector = sel+cclass;
8033     console.log('-- search style rule for '+selector)
8034
8035     for(let i = 0; i < rlen; i++){
8036         cr = ss.cssRules[i];
8037         console.log('CSS rule [' +i+'/' +rlen+' ] '+cr.selectorText);
8038         if( cr.selectorText === selector ){ // css class selector
8039             tabrule = ss.cssRules[i];
8040             console.log('CSS rule found for:[' +i+'/' +rlen+' ] '+selector);
8041             ss.deleteRule(i);
8042             //rlen = ss.cssRules.length;
8043             rulex = i
8044             // should search and replace the property here
8045         }
8046     }
8047     // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
8048     if( tabrule == null ){
8049         console.log('CSS rule NOT found for:[' +rlen+' ] '+selector);
8050         ss.insertRule(cstyle,rlen);
8051         ss.insertRule(cstyle,0); // override by 0?
8052         console.log('CSS rule inserted:[' +(rlen+1)+' ]\n'+cstyle);
8053     }else{
8054         ss.insertRule(cstyle,rlen);
8055         ss.insertRule(cstyle,0);
8056         console.log('CSS rule replaced:[' +(rlen+1)+' ]\n'+cstyle);
8057     }
8058 }
8059 function GJE_SetStyle(te){
8060     console.log('Apply the style to:'+te.id+'\n');

```

```

8061 console.log('Apply the style to:'+te.parentNode.id+'\n');
8062 console.log('Apply the style to:'+te.parentNode.class+'\n');
8063 cclass = te.parentNode.class;
8064 setCSSofClass(cclass,te.value); // should get selector part from
8065 // selector { rules }
8066
8067 if(false){
8068 //console.log('Apply the style:')
8069 //stid = this.parentNode.id+this.id+
8070 //stid = this.id+".style"
8071 css = te.value
8072 stid = te.parentNode.id+".style"
8073 e = document.getElementById(stid)
8074 if( e != null ){
8075 //console.log('Apply the style:'+e.id+'\n'+te.value);
8076 console.log('Apply the style:'+e.id+'\n'+css);
8077 // e.innerHTML = css; //te.value;
8078 //ncss = e.sheet;
8079 //ncss.insertRule(te.value,ncss.cssRules.length);
8080 }else{
8081 console.log('No element to Apply the style: '+stid)
8082 }
8083 tblid = te.parentNode.id+".table";
8084 e = document.getElementById(tblid);
8085 if( e != null ){
8086 //e.setAttribute('style',css);
8087 e.setProperty('style',css,'!important');
8088 }
8089 }
8090 }
8091 function makeTable(argv){
8092 //tid = ''
8093 cwe = GJE_CurElement
8094 tid = 'table_' + GJE_TableNo
8095
8096 nt = new Text('\n')
8097 cwe.appendChild(nt)
8098
8099 ne = document.createElement('span'); // the container
8100 cwe.appendChild(ne)
8101 ne.id = tid + '.span'
8102 ne.setAttribute('contenteditable',true)
8103
8104 htspan = document.createElement('span'); // html part
8105 //htspan.id = tid + '.html'
8106 //ne.innerHTML = '\n'
8107 nt = new Text('\n')
8108 ne.appendChild(nt)
8109 ne.appendChild(htspan)
8110
8111 htspan.id = tid
8112 htspan.setAttribute('class',tid)
8113
8114 ne.setAttribute('draggable','true')
8115 ne.addEventListener('drag',GJE_DragEvent);
8116 ne.addEventListener('dragend',GJE_DropEvent);
8117
8118 var col = 3
8119 var row = 2
8120 if( argv[0] != null ){
8121 col = argv[0]
8122 argv.shift()
8123 }
8124 if( argv[0] != null ){
8125 row = argv[0]
8126 argv.shift()
8127 }
8128
8129 //ne.setAttribute('class',tid)
8130 ht = "\n"
8131 //ht += '<'+table' + 'id="'+tid+'"' + ' class="'+tid+'"'
8132 ht += '<'+table'
8133 + ' onkeydown="GJE_TableKeyCommand(event,this)'"
8134 + ' ondrag="GJE_DragEvent(event,this)'\n'
8135 + ' ondragend="GJE_DropEvent(event,this)'\n'
8136 + ' draggable="true"\n'
8137 + ' contenteditable="true"'
8138 + '>\n'
8139 ht += '<'+tbody>\n';
8140 for( r = 0; r < row; r++ ){
8141 ht += "<"+tr>\n"
8142 for( c = 0; c < col; c++ ){
8143 ht += "<"+td>"
8144 ht += "ABCDEFGHILJKLNMNOPQRSTUVWXYZ".charAt(c) + r
8145 ht += "<"+td>\n"
8146 }
8147 ht += "<"+tr>\n"
8148 }
8149 ht += '<'+tbody>\n';
8150 ht += '<'+table>\n';
8151 htspan.innerHTML = ht;
8152 nt = new Text('\n')
8153 ne.appendChild(nt)
8154
8155 st = '#'+tid+' *{\n' // # for instanse specific
8156 + ' border:1px solid #aaa;\n'
8157 + ' background-color:#efe;\n'
8158 + ' color:#222;\n'
8159 + ' font-size:#14pt !important;\n'
8160 + ' font-family:monospace,Courier New !important;\n'
8161 + ' /* hit ESC to apply */'\n'
8162
8163 // wish script to be included
8164 //nj = document.createElement('script')
8165 //ne.appendChild(nj)
8166 //ne.innerHTML = 'function SetStyle(e){}'
8167
8168 // selector seems lost in dynamic style appending
8169 if(false){
8170 ns = document.createElement('style')
8171 ne.appendChild(ns)
8172 ns.id = tid + '.style'
8173 ns.innerHTML = '\n'+st
8174 nt = new Text('\n')
8175 ne.appendChild(nt)
8176 }
8177 setCSSofClass(tid,st); // should be in JavaScript script?
8178
8179 nx = document.createElement('textarea')
8180 ne.appendChild(nx)
8181 nx.id = tid + '-style_def'
8182 nx.setAttribute('class','GJ_StyleEditor')
8183 nx.spellcheck = false
8184 nx.cols = 60

```

```

8185     nx.rows = 10
8186     nx.innerHTML = '\n'+st
8187     nx.addEventListener('change',GJE_SetTableStyle);
8188     nx.addEventListener('keydown',GJE_StyleKeyCommand);
8189     //nx.addEventListener('click',GJE_SetTableStyle);
8190
8191     nt = new Text('\n')
8192     cwe.appendChild(nt)
8193
8194     GJE_TableNo += 1
8195     return 'created TABLE id="'+tid+'"'
8196 }
8197 function GJE_NodeEdit(argv){
8198     cwe = GJE_CurElement
8199     cmd = argv[0]
8200     argv.shift()
8201     args = argv.join(' ')
8202     ret = ""
8203
8204     if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
8205         if( GJE_NodeSaved != null ){
8206             xn = GJE_RootNode
8207             GJE_RootNode = GJE_NodeSaved
8208             GJE_NodeSaved = xn
8209             ret = '-- did undo'
8210         }else{
8211             ret = '-- could not undo'
8212         }
8213         return ret
8214     }
8215     GJE_NodeSaved = GJE_RootNode.cloneNode()
8216     if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
8217         if( argv[0] == null ){
8218             ne = GJE_RootNode
8219         }else
8220         if( argv[0] == '..' ){
8221             ne = cwe.parentNode
8222         }else{
8223             ne = document.getElementById(argv[0])
8224         }
8225         if( ne != null ){
8226             GJE_CurElement = ne
8227             ret = "-- current node: " + ne.id
8228         }else{
8229             ret = "-- not found: " + argv[0]
8230         }
8231     }else
8232     if( cmd == '.mkt' || cmd == '.mktable' ){
8233         makeTable(argv)
8234     }else
8235     if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
8236         ne = document.createElement(argv[0])
8237         //ne.id = argv[0]
8238         ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
8239         cwe.appendChild(ne)
8240         if( cmd == '.m' || cmd == '.mk' ){
8241             GJE_CurElement = ne
8242         }
8243     }else
8244     if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
8245         cwe.id = argv[0]
8246     }else
8247     if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
8248     }else
8249     if( cmd == '.h' || cmd == '.sh' || cmd == 'sh' ){
8250         s = argv.join(' ')
8251         cwe.innerHTML = s
8252     }else
8253     if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
8254         cwe.setAttribute(argv[0],argv[1])
8255     }else
8256     if( cmd == '.l' ){
8257     }else
8258     if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
8259         ret = cwe.innerHTML
8260     }else
8261     if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
8262         ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
8263         for( we = cwe.parentNode; we != null; ){
8264             ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
8265             we = we.parentNode
8266         }
8267     }else
8268     {
8269         ret = "Command: mk | rm \n"
8270         ret += "    pw -- print current node\n"
8271         ret += "    mk type -- make node with name and type\n"
8272         ret += "    nm name -- set the id #name of current node\n"
8273         ret += "    rm name -- remove named node\n"
8274         ret += "    cd name -- change current node\n"
8275     }
8276     //alert(ret)
8277     return ret
8278 }
8279 function GJC_Command(text){
8280     lines = text.value.split('\n')
8281     line = lines[lines.length-1]
8282     argv = line.split(' ')
8283     text.value += '\n'
8284     if( argv[0] == '#' ){ argv.shift() }
8285     args0 = argv.join(' ')
8286     cmd = argv[0]
8287     argv.shift()
8288     args = argv.join(' ')
8289
8290     if( cmd == 'nolog' ){
8291         StopConsoleLog = true
8292     }else
8293     if( cmd == 'new' ){
8294         if( argv[0] == 'table' ){
8295             argv.shift()
8296             console.log('argv'+argv)
8297             text.value += makeTable(argv)
8298         }else
8299         if( argv[0] == 'console' ){
8300             text.value += GJ_NewConsole('GJ_Console')
8301         }else{
8302             text.value += '-- new { console | table }'
8303         }
8304     }else
8305     if( cmd == 'strip' ){
8306         //text.value += GJF_StripClass()
8307     }else
8308     if( cmd == 'css' ){

```

```

8309     sel = '#table_1'
8310     if(argv[0]== '0')
8311         rule1 = sel+'{color:#000 !important; background-color:#fff !important;}';
8312     else
8313         rule1 = sel+'{color:#f00 !important; background-color:#eef !important;}';
8314     document.styleSheets[3].deleteRule(0);
8315     document.styleSheets[3].insertRule(rule1,0);
8316     text.value += 'CSS rule added: '+rule1
8317 }else
8318 if( cmd == 'print' ){
8319     e = null;
8320     if( e == null ){
8321         e = document.getElementById( 'GJFactory_0' )
8322     }
8323     if( e == null ){
8324         e = document.getElementById( 'GJFactory_1' )
8325     }
8326     if( argv[0] != null ){
8327         id = argv[0]
8328         if( id == 'f' ){
8329             //e = document.getElementById( 'GJE_RootNode' );
8330         }else{
8331             e = document.getElementById(id)
8332         }
8333         if( e != null ){
8334             text.value += e.outerHTML
8335         }else{
8336             text.value += "Not found: " + id
8337         }
8338     }else{
8339         text.value += GJE_RootNode.outerHTML
8340         //text.value += e.innerHTML
8341     }
8342 }else
8343 if( cmd == 'destroy' ){
8344     text.value += GJFactory_Destroy()
8345 }else
8346 if( cmd == 'save' ){
8347     e = document.getElementById( 'GJFactory' )
8348     Permanent.setItem( 'GJFactory-1', e.innerHTML )
8349     text.value += "-- Saved GJFactory"
8350 }else
8351 if( cmd == 'load' ){
8352     gjf = Permanent.getItem( 'GJFactory-1' )
8353     e = document.getElementById( 'GJFactory' )
8354     e.innerHTML = gjf
8355     // must restore EventListener
8356     text.value += "-- EventListener was not restored"
8357 }else
8358 if( cmd.charAt(0) == '.' ){
8359     argv0 = argv0.split(' ')
8360     text.value += GJE_NodeEdit(argv0)
8361 }else
8362 if( cmd == 'cont' ){
8363     bannerIsStopping = false
8364     GshMenuStop.innerHTML = "Stop"
8365 }else
8366 if( cmd == 'date' ){
8367     text.value += DateLong()
8368 }else
8369 if( cmd == 'echo' ){
8370     text.value += argv
8371 }else
8372 if( cmd == 'fork' ){
8373     html_fork()
8374 }else
8375 if( cmd == 'last' ){
8376     text.value += MyHistory
8377     //h = document.createElement("span")
8378     //h.innerHTML = MyHistory
8379     //text.value += h.innerHTML
8380     //tx = MyHistory.replace("\n","")
8381     //text.value += tx.replace("<"+>","<br>","\n") + "xxxx<"+>yyyy"
8382 }else
8383 if( cmd == 'ne' ){
8384     text.value += GJE_NodeEdit(argv)
8385 }else
8386 if( cmd == 'reload' ){
8387     location.reload()
8388 }else
8389 if( cmd == 'mem' ){
8390     text.value += GJC_Memory( 'GJC_Storage', argv, text )
8391 }else
8392 if( cmd == 'stop' ){
8393     bannerIsStopping = true
8394     GshMenuStop.innerHTML = "Start"
8395 }else
8396 if( cmd == 'who' ){
8397     text.value += "SessionId="+GJC_SessionId+" "+document.URL
8398 }else
8399 if( cmd == 'wall' ){
8400     text.value += GJC_Memory( 'GJC_Wall', 'write', text )
8401 }else
8402 {
8403     text.value += "Commands: help | echo | date | last \n"
8404     + '      new | save | load | mem \n'
8405     + '      who | wall | fork | nife'
8406 }
8407 }
8408
8409 function GJC_Input(){
8410     if( this.value.endsWith("\n") ){ // remove NL added by textarea
8411         this.value = this.value.slice(0,this.value.length-1)
8412     }
8413 }
8414
8415 var GCJ_Id = null
8416 function GJC_Resize(){
8417     GJC_Id.style.zIndex = 20000
8418     GJC_Id.style.width = window.innerWidth - 16
8419     GJC_Id.style.height = 300
8420     GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
8421     GJC_Id.style.color = "rgba(255,255,255,1.0)"
8422 }
8423 function GJC_FocusIn(){
8424     this.spellcheck = false
8425     SuppressGJShell = true
8426     this.onkeydown = GJC_Keydown
8427     GJC_Resize()
8428 }
8429 function GJC_FocusOut(){
8430     SuppressGJShell = false
8431     this.removeEventListener( 'keydown', GJC_Keydown );
8432 }

```

```

8433 window.addEventListener('resize',GJC_Resize);
8434
8435 function GJC_OnStorage(e){
8436     //alert('Got Message')
8437     //GJC.value += "\n((ReceivedMessage))\n"
8438 }
8439 window.addEventListener('storage',GJC_OnStorage);
8440 //window.addEventListener('storage',()=>{alert('GotMessage')})
8441
8442 function GJC_Setup(gjcId){
8443     gjcId.style.width = gsh.getBoundingClientRect().width
8444     gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
8445     //gjcId.value += "Date: " + DateLong() + "\n"
8446     gjcId.value += PS1
8447     gjcId.onfocus = GJC_FocusIn
8448     gjcId.addEventListener('input',GJC_Input);
8449     gjcId.addEventListener('focusout',GJC_FocusOut);
8450     GJC_Id = gjcId
8451 }
8452 function GJC_Clear(id){
8453 }
8454 if( document.getElementById("GJC_0") != null ){
8455     GJC_Setup(GJC_0)
8456 }else{
8457     document.write('<'+ 'textarea id="GJC_1" class="GJConsole"><'+ '/textarea>')
8458     GJC_Setup(GJC_1)
8459     factory = document.createElement('span');
8460     gsh.appendChild(factory)
8461     GJE_RootNode = factory;
8462     GJE_CurElement = GJE_RootNode;
8463 }
8464
8465 // TODO: focus handling
8466 </script>
8467 <style>
8468 .GJ_StyleEditor {
8469     font-size:9pt !important;
8470     font-family:Courier New, monospace !important;
8471 }
8472 </style>
8473
8474 </details>
8475 </span>
8476 <!-- ----- GJConsole END } ----- -->
8477 */
8478
8479 /*
8480 <span id="BlinderText">
8481 <style id="BlinderTextStyle">
8482 #GJLinkView {
8483     xxposition:absolute; z-index:5000;
8484     position:relative;
8485     display:block;
8486     left:8px;
8487     color:#fff;
8488     width:800px; height:300px; resize:both;
8489     margin:0px; padding:4px;
8490     background-color:rgba(200,200,200,0.5) !important;
8491 }
8492 .MsgText {
8493     width:578px !important;
8494     resize:both !important;
8495     color:#000 !important;
8496 }
8497 .GjNote {
8498     font-family:Georgia !important;
8499     font-size:13pt !important;
8500     color:#22a !important;
8501 }
8502 .textField {
8503     display:inline;
8504     border:0.5px solid #444;
8505     border-radius:3px;
8506     color:#000; background-color:#fff;
8507     width:106pt; height:18pt;
8508     margin:2px;
8509     padding:2px;
8510     resize:none;
8511     vertical-align:middle;
8512     font-size:10pt; font-family:Courier New;
8513 }
8514 .textLabel {
8515     border:0px solid #000 !important;
8516     background-color:rgba(0,0,0,0);
8517 }
8518 .textURL {
8519     width:300pt !important;
8520     border:0px solid #000 !important;
8521     background-color:rgba(0,0,0,0);
8522 }
8523 .VisibleText {
8524 }
8525 .BlinderText {
8526     color:#000; background-color:#eee;
8527 }
8528 .joinButton {
8529     font-family:Georgia !important;
8530     font-size:11pt;
8531     line-height:1.1;
8532     height:18pt;
8533     width:50pt;
8534     padding:3px !important;
8535     text-align:center !important;
8536     border-color:#aaa !important;
8537     border-radius:5px;
8538     color:#fff; background-color:#4a4 !important;
8539     vertical-align:middle !important;
8540 }
8541 .SendButton {
8542     vertical-align:top;
8543 }
8544 .ws0_log {
8545     font-size:10pt;
8546     color:#000 !important;
8547     line-height:1.0;
8548     background-color:rgba(255,255,255,0.7) !important;
8549     font-family:Courier New,monospace !important;
8550     width:99.3%;
8551     white-space:pre;
8552 }
8553 </style>
8554
8555 <!-- Form autofill test
8556 Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafirm/formLogin" size="80">

```

```

8557 <form method="POST" id="xxform" action="https://192.168.10.1/boafrm/FormLogin">
8558 dest? <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
8559 -->
8560 <details><summary>Form Auto. Filling</summary>
8561 <style>
8562 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
8563 display:inline !important; font-size:10pt !important; padding:1px !important;
8564 }
8565 </style>
8566 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
8567 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
8568 Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
8569 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
8570 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
8571 Sessionid:<input id="xxssid" class="xxinput" name="SESSION_ID" type="text" size="80">
8572 <input id="xxsubm" class="xxinput" type="submit" value="Submit"></form>
8573 </span>
8574 <script>
8575 function XXSetFormAction(){
8576     xxform.setAttribute('action',xxserv.value);
8577 }
8578 xxform.setAttribute('action',xxserv.value);
8579 xxserv.addEventListener('change',XXSetFormAction);
8580 //xxserv.value = location.href;
8581 </script>
8582 </details>
8583 */
8584
8585 /*
8586 <details id="BlinderTextClass" class="gsh-src"><summary>BlinderText</summary>
8587 <span id="BlinderTextScript">
8588 // https://w3c.github.io/uievents/#event-type-keydown
8589 //
8590 // 2020-09-21 class BlinderText - textarea element not to be readable
8591 //
8592 // BlinderText attributes
8593 // bl_plainText - null
8594 // bl_hideChecksum - [false]
8595 // bl_showLength - [false]
8596 // bl_visible - [false]
8597 // data-bl config - []
8598 // - min. length
8599 // - max. length
8600 // - acceptable charset in generate text
8601 //
8602 function BlinderChecksum(text){
8603     plain = text.bl_plainText;
8604     return strCRC32(plain,plain.length).toFixed(0);
8605 }
8606 function BlinderKeydown(ev){
8607     pass = ev.target
8608     if( ev.code == 'Enter' ){
8609         ev.preventDefault();
8610     }
8611     ev.stopPropagation()
8612 }
8613 function BlinderKeyUp1(ev){
8614     blind = ev.target
8615     if( ev.code == 'Backspace'){
8616         blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
8617     }else
8618     if( and(ev.code == 'KeyV', ev.ctrlKey) ){
8619         blind.bl_visible = !blind.bl_visible;
8620     }else
8621     if( and(ev.code == 'KeyL', ev.ctrlKey) ){
8622         blind.bl_showLength = !blind.bl_showLength;
8623     }else
8624     if( and(ev.code == 'KeyU', ev.ctrlKey) ){
8625         blind.bl_plainText = "";
8626     }else
8627     if( and(ev.code == 'KeyR', ev.ctrlKey) ){
8628         checksum = BlinderChecksum(blind);
8629         blind.bl_plainText = checksum; //.toString(32);
8630     }else
8631     if( ev.code == 'Enter' ){
8632         ev.stopPropagation();
8633         ev.preventDefault();
8634         return;
8635     }else
8636     if( ev.key.length != 1 ){
8637         console.log('KeyUp: '+ev.code+'/'+ev.key);
8638         return;
8639     }else{
8640         blind.bl_plainText += ev.key;
8641     }
8642
8643     leng = blind.bl_plainText.length;
8644     //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
8645     checksum = BlinderChecksum(blind) % 10; // show last one digit only
8646
8647     visual = '';
8648     if( !blind.bl_hideChecksum || blind.bl_showLength ){
8649         visual += '[';
8650     }
8651     if( !blind.bl_hideChecksum ){
8652         visual += '#'+checksum.toString(10);
8653     }
8654     if( blind.bl_showLength ){
8655         visual += '/' + leng;
8656     }
8657     if( !blind.bl_hideChecksum || blind.bl_showLength ){
8658         visual += ']' ;
8659     }
8660     if( blind.bl_visible ){
8661         visual += blind.bl_plainText;
8662     }else{
8663         visual += '*'.repeat(leng);
8664     }
8665     blind.value = visual;
8666 }
8667 function BlinderKeyUp(ev){
8668     BlinderKeyUp1(ev);
8669     ev.stopPropagation();
8670 }
8671 // https://w3c.github.io/uievents/#keyboardevent
8672 // https://w3c.github.io/uievents/#uievent
8673 // https://dom.spec.whatwg.org/#event
8674 function BlinderTextEvent(){
8675     ev = event;
8676     blind = ev.target;
8677     console.log('Event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
8678     if( ev.type == 'keyup' ){
8679         BlinderKeyUp(ev);
8680     }else

```

```

8681     if( ev.type == 'keydown' ){
8682         BlinderKeydown(ev);
8683     }else{
8684         console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
8685     }
8686 }
8687 //< textarea hidden id="BlinderTextClassDef" class="textField"
8688 // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
8689 // spellcheck="false">< /textarea>
8690 //< textarea hidden id="gj_pass1"
8691 // class="textField BlinderText"
8692 // placeholder="PassWord1"
8693 // onkeydown="BlinderTextEvent()"
8694 // onkeyup="BlinderTextEvent()"
8695 // spellcheck="false">< /textarea>
8696 function SetupBlinderText(parent,txa,phold){
8697     if( txa == null ){
8698         txa = document.createElement('textarea');
8699         //txa.id = id;
8700     }
8701     txa.setAttribute('class','textField BlinderText');
8702     txa.setAttribute('placeholder',phold);
8703     txa.setAttribute('onkeydown','BlinderTextEvent()');
8704     txa.setAttribute('onkeyup','BlinderTextEvent()');
8705     txa.setAttribute('spellcheck','false');
8706     //txa.setAttribute('bl_plainText','false');
8707     txa.bl_plainText = '';
8708     //parent.appendChild(txa);
8709 }
8710 function DestroyBlinderText(txa){
8711     txa.removeAttribute('class');
8712     txa.removeAttribute('placeholder');
8713     txa.removeAttribute('onkeydown');
8714     txa.removeAttribute('onkeyup');
8715     txa.removeAttribute('spellcheck');
8716     txa.bl_plainText = '';
8717 }
8718 //
8719 // visible textarea like Username
8720 //
8721 function VisibleTextEvent(){
8722     if( event.code == 'Enter' ){
8723         if( event.target.NoEnter ){
8724             event.preventDefault();
8725         }
8726     }
8727     event.stopPropagation();
8728 }
8729 function SetupVisibleText(parent,txa,phold){
8730     if( false ){
8731         txa.setAttribute('class','textField VisibleText');
8732     }else{
8733         newclass = txa.getAttribute('class');
8734         if( and(newclass != null, newclass != '') ){
8735             newclass += ' ';
8736         }
8737         newclass += 'VisibleText';
8738         txa.setAttribute('class',newclass);
8739     }
8740     //console.log('SetupVisibleText class='+txa.class);
8741     txa.setAttribute('placeholder',phold);
8742     txa.setAttribute('onkeydown','VisibleTextEvent()');
8743     txa.setAttribute('onkeyup','VisibleTextEvent()');
8744     txa.setAttribute('spellcheck','false');
8745     cols = txa.getAttribute('cols');
8746     if( cols != null ){
8747         txa.style.width = '580px';
8748         //console.log('VisualText#'+txa.id+' cols='+cols)
8749     }else{
8750         //console.log('VisualText#'+txa.id+' NO cols')
8751     }
8752     rows = txa.getAttribute('rows');
8753     if( rows != null ){
8754         txa.style.height = '30px';
8755         txa.style.resize = 'both';
8756         txa.NoEnter = false;
8757     }else{
8758         txa.NoEnter = true;
8759     }
8760 }
8761 function DestroyVisibleText(txa){
8762     txa.removeAttribute('class');
8763     txa.removeAttribute('placeholder');
8764     txa.removeAttribute('onkeydown');
8765     txa.removeAttribute('onkeyup');
8766     txa.removeAttribute('spellcheck');
8767     cols = txa.removeAttribute('cols');
8768 }
8769 </span>
8770 <script>
8771 js = document.getElementById('BlinderTextScript');
8772 eval(js.innerHTML);
8773 //js.outerHTML = ""
8774 </script>
8775
8776 </details>
8777 </span>
8778 */
8779 /*
8780 <script id="GJLinkScript">
8781 function gjkey_hash(text){
8782     return strCRC32(text,text.length) % 0x10000;
8783 }
8784 function gj_addlog(e,msg){
8785     now = (new Date().getTime() / 1000).toFixed(3);
8786     tstp = '['+now+' ]
8787     e.value += tstp + msg;
8788     e.scrollTop = e.scrollHeight;
8789 }
8790 }
8791 function gj_addlog_cl(msg){
8792     ws0_log.value += '(console.log) ' + msg + '\n';
8793 }
8794 var GJ_Channel = null;
8795 var GJ_Log = null;
8796 var gjx; // the global variable
8797 function GJ_Join(){
8798     target = gj_join;
8799     if( target.value == 'Leave' ){
8800         GJ_Channel.close();
8801         GJ_Channel = null;
8802         target.value = 'Join';
8803     }
8804 }

```

```

8805
8806 var ws0;
8807 var ws0_log;
8808
8809 sav_console_log = console.error
8810 console.error = gj_addlog_cl
8811 ws0 = new WebSocket(gj_serv.innerHTML);
8812 console.error = sav_console_log
8813
8814 GJ_Channel = ws0;
8815 ws0_log = document.getElementById('ws0_log');
8816 GJ_Log = ws0_log;
8817
8818 now = (new Date().getTime() / 1000).toFixed(3);
8819 const wsstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];
8820 cst = wsstats[ws0.readyState];
8821 gj_addlog(ws0_log,'stat '+ws0.readyState+'('+cst+') : GJ Linked\n');
8822
8823 ws0.addEventListener('error', function(event){
8824     gj_addlog(ws0_log,'stat error : transport error?\n');
8825 });
8826 ws0.addEventListener('open', function(event){
8827     GJLinkView.style.zIndex = 10000;
8828     //console.log('#'+GJLinkView.id+'.zIndex='+GJLinkView.style.zIndex);
8829     date1 = new Date().getTime();
8830     date2 = (date1 / 1000).toFixed(3);
8831     seed = date1.toString(16);
8832
8833     // user name and key
8834     user = document.getElementById('gj_user').value;
8835     if( user.length == 0 ){
8836         gj_user.value = 'nemo';
8837         user = 'nemo';
8838     }
8839     key1 = document.getElementById('gj_ukey').bl_plainText;
8840     ukey = gjkey_hash(seed+user+key1).toString(16);
8841
8842     // session name and key
8843     chan = document.getElementById('gj_chan').value;
8844     if( chan.length == 0 ){
8845         gj_chan.value = 'main';
8846         chan = 'main';
8847     }
8848     key2 = document.getElementById('gj_ckey').bl_plainText;
8849     ckey = gjkey_hash(seed+chan+key2).toString(16);
8850
8851     msg = date2 + ' JOIN ' + user + '|' + chan + ' ' + ukey + ':' + ckey;
8852     gj_addlog(ws0_log,'send '+msg+'\n');
8853     ws0.send(msg);
8854
8855     target.value = 'Leave';
8856     //console.log('[ '+date2+' ] #'+target.id+' '+target.value+'\n');
8857     //gj_addlog(ws0_log,'label '+target.value+'\n');
8858 });
8859 ws0.addEventListener('message', function(event){
8860     now = (new Date().getTime() / 1000).toFixed(3);
8861     msg = event.data;
8862     gj_addlog(ws0_log,'recv '+msg+'\n');
8863
8864     argv = msg.split(' ');
8865     tstamp = argv[0];
8866     argv.shift();
8867     if( argv[0] == 'reload' ){
8868         location.reload()
8869     }
8870     argv.shift(); // command
8871     argv.shift(); // from/to
8872     if( argv[0] == 'auth' ){
8873         // doing authorization required
8874     }
8875     if( argv[0] == 'echo' ){
8876         now = (new Date().getTime() / 1000).toFixed(3);
8877         msg = now + ' RESP ' + argv.join(' ');
8878         gj_addlog(ws0_log,'send '+msg+'\n');
8879         ws0.send(msg);
8880     }
8881     if( argv[0] == 'eval' ){
8882         argv.shift();
8883         js = argv.join(' ');
8884         ret = eval(js); // <----- eval()
8885         gj_addlog(ws0_log,'eval '+js+' '+ret+'\n');
8886         now = (new Date().getTime() / 1000).toFixed(3);
8887         msg = now + ' ' + 'RESP' + ret;
8888         ws0.send(msg);
8889         gj_addlog(ws0_log,'send '+msg+'\n')
8890     }
8891 });
8892 ws0.addEventListener('close', function(event){
8893     if( GJ_Channel == null ){
8894         gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
8895         return;
8896     }
8897     GJ_Channel.close();
8898     GJ_Channel = null;
8899     target.value = 'Join';
8900     gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
8901 });
8902 }
8903 function GJ_SendMessageUserPass(user,chan,msgbody){
8904     now = (new Date().getTime() / 1000).toFixed(3);
8905     msg = now + ' ISAY '+user+'|'+chan+' '+msgbody;
8906     gj_addlog(GJ_Log,'send '+msg+'\n');
8907     GJ_Channel.send(msg);
8908 }
8909 function GJ_SendMessage(msgbody){
8910     if( GJ_Channel == null ){
8911         gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
8912         return;
8913     }
8914     //target = event.target;
8915     user = document.getElementById('gj_user').value;
8916     chan = document.getElementById('gj_chan').value;
8917     GJ_SendMessageUserPass(user,chan,msgbody);
8918 }
8919 function GJ_Send(){
8920     msgbody = gj_sendText.value;
8921     GJ_SendMessage(msgbody);
8922 }
8923 </script>
8924
8925 <!-- ----- GJLINK ----- -->
8926 <!--
8927 - User can subscribe to a channel
8928 - A channel will be broadcasted

```



```

8929 - A channel can be a pattern (regular expression)
8930 - User is like From:(me) and channel is like To: or Recipient:
8931 - like VIABUS
8932 - watch message with SENDME, WATCH, CATCH, HEAR, or so
8933 - routing with path expression or name pattern (with routing with DNS like system)
8934 -->
8935 */
8936
8937 <<span id="GJLinkGolang">
8938 <<details id="GshWebSocket" class="gsh-src"><summary>Golang / JavaScript Link</summary>
8939 // 2020-0920 created
8940 <<a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
8941 <<a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
8942 // INSTALL: go get golang.org/x/net/websocket
8943 // INSTALL: sudo {apt,yum} install git (if git is not installed yet)
8944 // import "golang.org/x/net/websocket"
8945 const gshws_origin = "http://localhost:9999"
8946 const gshws_server = "localhost:9999"
8947 const gshws_port = 9999
8948 const gshws_path = "gjlink1"
8949 const gshws_url = "ws://" + gshws_server + "/" + gshws_path
8950 const GSHWS_MSGSIZE = (8*1024)
8951 func fmtstring(fmts string, params ...interface{})(string){
8952     return fmt.Sprintf(fmts,params...)
8953 }
8954 func GSHWS_MARK(what string)(string){
8955     now := time.Now()
8956     us := fmtstring("%06d",now.Nanosecond() / 1000)
8957     mark := ""
8958     if( !AtConsoleLineTop ){
8959         mark += "\n"
8960         AtConsoleLineTop = true
8961     }
8962     mark += "[" + now.Format(time.Stamp) + "." + us + " ] -GJ- " + what + " : "
8963     return mark
8964 }
8965 func gchk(what string,err error){
8966     if( err != nil ){
8967         panic(GSHWS_MARK(what)+err.Error())
8968     }
8969 }
8970 func glog(what string, fmts string, params ...interface{}){
8971     fmt.Print(GSHWS_MARK(what))
8972     fmt.Printf(fmts+"\n",params...)
8973 }
8974
8975 var WSV = []*websocket.Conn{}
8976 func jsend(argv []string){
8977     if len(argv) <= 1 {
8978         fmt.Printf("--Ij %v [-m] command arguments\n",argv[0])
8979         return
8980     }
8981     argv = argv[1:]
8982     if( len(WSV) == 0 ){
8983         fmt.Printf("--Ej-- No link now\n")
8984         return
8985     }
8986     if( 1 < len(WSV) ){
8987         fmt.Printf("--Ij-- multiple links (%v)\n",len(WSV))
8988     }
8989
8990     multicast := false // should be filtered with regexp
8991     if( 0 < len(argv) && argv[0] == "-m" ){
8992         multicast = true
8993         argv = argv[1:]
8994     }
8995     args := strings.Join(argv, " ")
8996
8997     now := time.Now()
8998     msec := now.UnixNano() / 1000000;
8999     tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9000     msg := fmtstring("%v SEND gshell|* %v",tstamp,args)
9001
9002     if( multicast ){
9003         for i,ws := range WSV {
9004             wn,werr := ws.Write([]byte(msg))
9005             if( werr != nil ){
9006                 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
9007             }
9008             glog("SQ",fmtstring("(%v) %v",wn,msg))
9009         }
9010     }else{
9011         i := 0
9012         ws := WSV[i]
9013         wn,werr := ws.Write([]byte(msg))
9014         if( werr != nil ){
9015             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
9016         }
9017         glog("SQ",fmtstring("(%v) %v",wn,msg))
9018     }
9019 }
9020 func servl(ws *websocket.Conn) {
9021     WSV = append(WSV,ws)
9022     //fmt.Print("\n")
9023     glog("CO","accepted connections[%v]",len(WSV))
9024     //remoteAddr := ws.RemoteAddr
9025     //fmt.Printf("-- accepted %v\n",remoteAddr)
9026     //fmt.Printf("-- accepted %v\n",ws.Config())
9027     //fmt.Printf("-- accepted %v\n",ws.Config().Header)
9028     //fmt.Printf("-- accepted %v // %v\n",ws,servl)
9029
9030     var reqb = make([]byte,GSHWS_MSGSIZE)
9031     for {
9032         rn, rerr := ws.Read(reqb)
9033         if( rerr != nil || rn < 0 ){
9034             glog("SQ",fmtstring("(%v,%v)",rn,rerr))
9035             break
9036         }
9037         req := string(reqb[0:rn])
9038         glog("SQ",fmtstring("(%v) %v",rn,req))
9039
9040         margv := strings.Split(req, " ");
9041         margv = margv[1:]
9042         if( 0 < len(margv) ){
9043             if( margv[0] == "RESP" ){
9044                 // should forward to the destination
9045                 continue;
9046             }
9047         }
9048         now := time.Now()
9049         msec := now.UnixNano() / 1000000;
9050         tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9051         res := fmtstring("%v "+CAST+" %v",tstamp,req)
9052         wn, werr := ws.Write([]byte(res))

```

```

9053     gchk("SE",werr)
9054     glog("SR",fmtstring("(%v) %v",wn,string(res)))
9055     }
9056     glog("SF","WS response finish")
9057
9058     wsv := []*websocket.Conn{}
9059     wsx := 0
9060     for i,v := range WSV {
9061         if( v != ws ){
9062             wsx = i
9063             wsv = append(wsv,v)
9064         }
9065     }
9066     WSV = wsv
9067     //glog("CO","closed %v",ws)
9068     glog("CO","closed connection [%v/%v]",wsx+1,len(WSV)+1)
9069     ws.Close()
9070 }
9071 // url ::= [scheme://]host[:port][/path]
9072 func decomp_URL(url string){
9073
9074     func full_wsURL(){
9075     }
9076     func gj_server(argv []string) {
9077         gjserv := gshws_url
9078         gjport := gshws_server
9079         gjpath := gshws_path
9080         gjscheme := "ws"
9081
9082         //cmd := argv[0]
9083         argv = argv[1:]
9084         if( 1 <= len(argv) ){
9085             serv := argv[0]
9086             if( 0 < strings.Index(serv,"://") ){
9087                 schemev := strings.Split(serv,"://")
9088                 gjscheme = schemev[0]
9089                 serv = schemev[1]
9090             }
9091             if( 0 < strings.Index(serv,"/") ){
9092                 pathv := strings.Split(serv,"/")
9093                 serv = pathv[0]
9094                 gjpath = pathv[1]
9095             }
9096             servv := strings.Split(serv,":")
9097             host := "localhost"
9098             port := 9999
9099             if( servv[0] != "" ){
9100                 host = servv[0]
9101             }
9102             if( len(servv) == 2 ){
9103                 fmt.Sscanf(servv[1],"%d",&port)
9104             }
9105             //glog("LC","hostport=%v (%v : %v)",servv,host,port)
9106             gjport = fmt.Sprintf("%v:%v",host,port)
9107             gjserv = gjscheme + "://" + gjport + "/" + gjpath
9108         }
9109         glog("LS",fmtstring("listening at %v",gjserv))
9110         http.Handle("/"+gjpath,websocket.Handler(serv))
9111         err := error(nil)
9112         if( gjscheme == "wss" ){
9113             // https://golang.org/pkg/net/http/#ListenAndServeTLS
9114             //err = http.ListenAndServeTLS(gjport,nil)
9115         }else{
9116             err = http.ListenAndServe(gjport,nil)
9117         }
9118         gchk("LE",err)
9119     }
9120
9121     func gj_client(argv []string) {
9122         glog("CS",fmtstring("connecting to %v",gshws_url))
9123         ws, err := websocket.Dial(gshws_url,"",gshws_origin)
9124         gchk("C",err)
9125
9126         var resb = make([]byte, GSHWS_MSGSIZE)
9127         for qi := 0; qi < 3; qi++ {
9128             req := fmtstring("Hello, GShell! (%v)",qi)
9129             wn, werr := ws.Write([]byte(req))
9130             glog("QM",fmtstring("(%v) %v",wn,req))
9131             gchk("QE",werr)
9132             rn, rerr := ws.Read(resb)
9133             gchk("RE",rerr)
9134             glog("RM",fmtstring("(%v) %v",rn,string(resb)))
9135         }
9136         glog("CF","WS request finish")
9137     }
9138     //</details></span>
9139     /*
9140     <details><summary>GJ Link</summary>
9141     <span id="GJLinkView" class="GJLinkView">
9142     <p>
9143     <note class="GJNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
9144     </p>
9145     <p>
9146     <span id="GJLink 1">
9147     <script id="gj_xxx1_gen">
9148     if( document.getElementById('gj_serv') == null ){ // executed twice??
9149         document.write('<'+span id="gj_serv_label" class="textField textLabel">Server: <'+/span>');
9150         document.write('<'+span id="gj_serv" class="textField textURL" contenteditable><'+/span>');
9151     }
9152     </script>
9153     <br>
9154     <input id="gj_join" type="button" class="joinButton onclick="GJ_Join()" value="Join">
9155     <script id="gj_xxx2_gen">
9156     if( true ){
9157         document.write('<'+textarea id="gj_user" class="textField"><'+/textarea>');
9158         document.write('<'+textarea id="gj_ukey" class="textField"><'+/textarea>');
9159         document.write('<'+textarea id="gj_chan" class="textField"><'+/textarea>');
9160         document.write('<'+textarea id="gj_cke" class="textField"><'+/textarea>');
9161     }
9162     </script>
9163     <br>
9164     <input id="gj_sendButton" type="button" class="joinButton SendButton onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
9165     <script id="gj_sendText_gen">
9166     if( true ){
9167         document.write('<'+textarea id="gj_sendText" class="textField MssgText" cols=60 rows=2><'+/textarea>');
9168     }
9169     </script>
9170     </span></p>
9171     <p>
9172     <script id="ws0_log_gen">
9173     if( true ){
9174         document.write('<'+textarea id="ws0_log" class="ws0_log"
9175         + ' cols=100 rows=10 spellcheck="false"><'+/textarea>');
9176     }

```

```

9177 }
9178 </script>
9179 </p>
9180 </span>
9181 <script>
9182 function SetupGJLink(){
9183     SetupVisibleText(GJLink_1,gj_serv,'GJLinkSv');
9184     SetupVisibleText(GJLink_1,gj_user,'UserName');
9185     SetupBlinderText(GJLink_1,gj_ukey,'UserKey');
9186     SetupVisibleText(GJLink_1,gj_chan,'ChannelName');
9187     SetupBlinderText(GJLink_1,gj_ckey,'ChannelKey');
9188     SetupVisibleText(GJLink_1,gj_sendText,'Message');
9189     gj_serv.innerHTML = 'ws://localhost:9999/gjlink1'
9190 }
9191 SetupGJLink();
9192 function iselem(eid){
9193     return document.getElementById(eid);
9194 }
9195 function DestroyGJLink1(){
9196     if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
9197     if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
9198     if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
9199     if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
9200     if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
9201     if( iselem('gj_ckekey') ) gj_ckekey.parentNode.removeChild(gj_ckekey);
9202     if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
9203     if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
9204 }
9205 DestroyGJLink = DestroyGJLink1;
9206 </script>
9207 </details>
9208 */
9209
9210 /*
9211 <script id="HtmlCodeview-script">
9212     function showHtmlCode(otxa,code){
9213         if( event.target.value == 'ShowCode' ){
9214             txa = document.createElement('textarea');
9215             txa.id = otxa.id;
9216             txa.setAttribute('class','HtmlCodeviewText');
9217             otxa.parentNode.replaceChild(txa,otxa);
9218             txa.setAttribute('spellcheck','false');
9219             txa.value = code.innerHTML;
9220             txa.style.display = "block";
9221             txa.style.width = "100%";
9222             txa.style.height = "300px";
9223             event.target.value = 'HideCode';
9224         }else{
9225             otxa.style.display = "none";
9226             event.target.value = 'ShowCode';
9227         }
9228     }
9229 </script>
9230 <style id="HtmlCodeview-style">
9231     .HtmlCodeviewText {
9232         font-size:10pt;
9233         font-family:Courier New;
9234         white-space:pre;
9235     }
9236     .HtmlCodeviewButton {
9237         font-size:11pt;
9238         line-height:1.2;
9239         font-family:Georgia;
9240         border-radius:3px;
9241         color:#ddd; background-color:#333;
9242     }
9243 </style>
9244 */
9245
9246 /*
9247 <details><summary>Live HTML Snapshot</summary>
9248 <span id="LiveHTML">
9249 <!-- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
9250 <input class="HtmlCodeviewButton" type="button" value="ShowCode" onclick="showLiveHTMLCode()">
9251 <span id="LiveHTML_Codeview"></span>
9252 <script id="LiveHTMLScript">
9253 function showLiveHTMLCode(){
9254     showHtmlCode(LiveHTML_Codeview,LiveHTML);
9255 }
9256 var _editable = false;
9257 var savSuppressGJShell = false;
9258 function ToggleEditMode(){
9259     _editable = !_editable;
9260     if( _editable ){
9261         savSuppressGJShell = SuppressGJShell;
9262         SuppressGJShell = true;
9263         gsh.setAttribute('contenteditable','true');
9264         GshMenuEdit.innerHTML = 'Lock';
9265         GshMenuEdit.style.color = 'rgba(255,0,0,1)';
9266         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
9267     }else{
9268         SuppressGJShell = savSuppressGJShell;
9269         gsh.setAttribute('contenteditable','false');
9270         GshMenuEdit.innerHTML = 'Edit';
9271         GshMenuEdit.style.color = 'rgba(16,160,16,1)';
9272         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
9273     }
9274 }
9275 function html_edit(){
9276     ToggleEditMode();
9277 }
9278
9279 // Live HTML (DOM) Snapshot onto browser's localStorage
9280 // 2020-0923 SatoxITS
9281 var htRoot = gsh // -- Element-ID, should be selectable
9282 const snappedHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStogate
9283 // -- should be a [map] of URL
9284 // -- should be with CSSOM as inline script
9285 const htVersionTag = 'VersionTag'; // VesionTag Element-ID in the HTML (in DOM)
9286 function showVersion(note,v,w,u,t){
9287     w.alert(note+' : ' + v + '\n'
9288         + '-- URL: ' + u + '\n'
9289         + '-- Time: ' + DateLong0(t*1000)
9290     );
9291 }
9292 function html_save(){
9293     u = document.URL;
9294     t = new Date().getTime() / 1000;
9295     v = '<'+span id="+htVersionTag+" data-url="+u+" data-time="+t+">';
9296     v += '<'+/span>\n';
9297     h += v + htRoot.outerHTML;
9298     localStorage.setItem(snappedHTML,h);
9299     showVersion("Saved",window,v,u,t);
9300 }

```

```

9301 function html_load(){
9302     h = localStorage.getItem(snappedHTML);
9303     if( h == null ){
9304         alert('No snapshot taken yet');
9305         return;
9306     }
9307     w = window.open('', '', '');
9308     d = w.document;
9309     d.write(h);
9310     w.focus();
9311     html_ver1("Loaded",w,d);
9312 }
9313 function html_ver1(note,w,d){
9314     if( (v = d.getElementById(htVersionTag)) != null ){
9315         h = v.outerHTML;
9316         u = v.getAttribute('data-url');
9317         t = v.getAttribute('data-time');
9318     }else{
9319         h = 'No version info. in the page';
9320         u = '';
9321         t = 0;
9322     }
9323     showVersion(note,w,v,u,t);
9324 }
9325 function html_ver0(){
9326     html_ver1("Version",window,document);
9327 }
9328 </script>
9329 <!-- LiveHTML -->
9330 </span>
9331 </details>
9332 */
9333
9334 /*
9335 <details><summary>Event sharing</summary>
9336 <span id="EventSharingCodeSpan">
9337
9338 <!-- ----- Event sharing // 2020-0925 SatoxITS { -->
9339
9340 <div id="iftestTemplate" class="iftest" hidden="">
9341 <style>_iftestbody{ color:#f22;font-family:Georgia;font-size:10pt;} </style>
9342 <span id="frameBody" class="iftestbody" onclick="frameClick()"><script>
9343     function docadd(txt){
9344         document.body.append(txt);
9345         window.scrollTo(0,100000);
9346     }
9347     function frameClick(){
9348         xy = '(x='+event.x + ' y='+event.y+')';
9349         //docadd('Got Click on #' +event.target.id+ ' +xy+ '\n');
9350         docadd('Got Click on #' +Fid.value+ ' +xy+ '\n');
9351         window.scrollTo(0,100000);
9352         window.parent.postMessage('OnClick: '+xy,'*');
9353     }
9354     function frameMouseMove(){
9355         if( false ){
9356             document.body.append('Mousemove on #' +event.target.id+ '
9357             + 'x='+event.x + ' y='+event.y + '\n');
9358             peerWin = window.frames.iframe1;
9359             document.body.append('Send to peer #' +peerWin+ ' + '\n');
9360             window.scrollTo(0,100000);
9361             peerWin.postMessage('Hi!', '*');
9362         }
9363     }
9364     function frameKeydown(){
9365         msg = 'Got Keydown: #' +Fid.value+ ' (' +event.code+ ')';
9366         docadd(msg+'\n');
9367         window.parent.postMessage(msg,'*');
9368     }
9369     function frameOnMessage(){
9370         docadd('Message ' + event.data + '\n');
9371         window.scrollTo(0,100000);
9372     }
9373     if( document.getElementById('Fid') ){
9374         frameBody.id = Fid.value;
9375         h = '';
9376         h += '<'+style>*<';
9377         h += 'font-size:10pt;white-space:pre-wrap;';
9378         h += 'font-family:Courier New;';
9379         h += '<'+style>';
9380         h += 'I am '+Fid.value+'\n';
9381         document.write(h);
9382         window.addEventListener('click',frameClick);
9383         window.addEventListener('keydown',frameKeydown);
9384         window.addEventListener('message',frameOnMessage);
9385         window.addEventListener('mousemove',frameMouseMove);
9386         window.parent.postMessage('Hi parent, I am '+Fid.value,'*');
9387     }
9388 </script></span></div>
9389
9390 <style>.iframeTest{margin:3px;resize:both;width:370px;height:120px;}</style>
9391 <h2>Inter-window communicaiton</h2>
9392 <note>
9393 frame0 >>> frame1 and frame2<br>
9394 frame1 >>> frame0 and frame2<br>
9395 frame2 >>> frame0 and frame1<br>
9396 </note>
9397 <div id="iframe-test">
9398 <pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
9399 <iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
9400 <iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
9401 <iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
9402 </div>
9403
9404 <script id="if0-test-script">
9405     setupFrames0();
9406     setupFrames12();
9407
9408     function setFrameSrcdoc(dst,src){
9409         if( true ){
9410             dst.contentWindow.document.write(src);
9411             // this makes browser waite close, and crash if accumulated !?
9412             // so it should be closed after write
9413             dst.contentWindow.document.close();
9414         }else{
9415             // to be erased before source dump
9416             // but shold be set for live snapshot
9417             dst.srcdoc = src;
9418         }
9419     }
9420     function setupFrames0(){
9421         ibody = iframe0.contentWindow.document.body;
9422         iframe0.style.width = "755px"
9423         //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
9424         window.addEventListener('message',messageFromChild);

```

```

9425
9426     if0 = '';
9427     if0 += '<'+pre style="font-family:Courier New;">';
9428     if0 += '<input id="Pid" value="iframe0">';
9429     if0 += iftestTemplate.innerHTML;
9430     setFrameSrcdoc(iframe0,if0);
9431
9432     function clickOnChild(){
9433         console.log('clickOn #' +this.id);
9434     }
9435     function moveOnChild(){
9436         console.log('moveOn #' +this.id);
9437     }
9438     iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
9439     iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
9440 }
9441 function setupFrames12(){
9442     if1 = '<input id="Pid" value="iframe1">';
9443     if1 += iftestTemplate.innerHTML;
9444     setFrameSrcdoc(iframe1,if1);
9445     //iframe1.name = 'iframe1'; // this seems break contentWindow
9446
9447     if2 = '<input id="Pid" value="iframe2">';
9448     if2 += iftestTemplate.innerHTML;
9449     setFrameSrcdoc(iframe2,if2);
9450
9451     iframe1.addEventListener('message',messageFromChild);
9452     //iframe1.addEventListener('mouseover',moveOnChild);
9453     iframe2.addEventListener('message',messageFromChild);
9454     //iframe2.addEventListener('mouseover',moveOnChild);
9455     iframe1.contentWindow.postMessage('parent0 Hi iframe1 -- from parent.','*');
9456     //iframe1.contentWindow.postMessage('Your peer is '+iframe2.contentWindow,'*');
9457     iframe2.contentWindow.postMessage('parent0 Hi iframe2 -- from parent.','*');
9458     //iframe2.contentWindow.postMessage('Your peer is '+iframe1.contentWindow,'*');
9459 }
9460 function messageFromChild(){
9461     from = null;
9462     forw = null;
9463     if( event.source == iframe0.contentWindow ){
9464         from = 'iframe0'
9465         forw = 'iframe12';
9466     }else
9467     if( event.source == iframe1.contentWindow ){
9468         from = 'iframe1'
9469         forw = 'iframe2';
9470     }else
9471     if( event.source == iframe2.contentWindow ){
9472         from = 'iframe2'
9473         forw = 'iframe1';
9474     }else
9475     {
9476         iframeHost.innerHTML += 'Message [unknown] '
9477         + ' orig=' + event.origin
9478         + ' data=' + event.data
9479         //+ ' from=' + event.source
9480         ;
9481     }
9482     msglog1 = from + event.data + ' -- '
9483     + ' from=' + event.source
9484     + ' orig=' + event.origin
9485     + ' name=' + event.source.name
9486     //+ ' port=' + event.ports
9487     //+ ' evid=' + event.lastEventId
9488     + '\n'
9489     ;
9490     if( true ){
9491         if( forw == 'iframe1' || forw == 'iframe12' ){
9492             iframe1.contentWindow.postMessage(from+event.data);
9493         }
9494         if( forw == 'iframe2' || forw == 'iframe12' ){
9495             iframe2.contentWindow.postMessage(from+event.data);
9496         }
9497     }
9498     txtadd0(msglog1);
9499
9500     function txtadd0(txt){
9501         iframe0.contentWindow.document.body.append(txt);
9502         iframe0.contentWindow.scrollTo(0,100000);
9503     }
9504 }
9505 function es_ShowSelf(){
9506     iframe1.setAttribute('src',document.URL);
9507     iframe2.setAttribute('src',document.URL);
9508 }
9509 </script>
9510
9511 <input class="HtmlCodeviewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
9512 <input class="HtmlCodeviewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
9513 <span id="EventSharingCodeview"><span>
9514 <script id="EventSharingScript">
9515 function es_showHtmlCode(){
9516     showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
9517 }
9518 DestroyEventSharingCodeview = function(){
9519     //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
9520     EventSharingCodeview.innerHTML = "";
9521     iframe0.style = "";
9522     //iframe0.srcdoc = "erased";
9523     //iframe1.srcdoc = "erased";
9524     //iframe2.srcdoc = "erased";
9525 }
9526 </script>
9527 <!-- EventSharing -->
9528 </span>
9529 </details>
9530 */
9531 */
9532 /*
9533 <!-- ----- "GShell Inside" Notification { -->
9534 <script id="script-gshell-inside">
9535 var notices = 0;
9536 function noticeGShellInside(){
9537     ver = '';
9538     if( ver = document.getElementById('GshVersion') ){
9539         ver = ver.innerHTML;
9540     }
9541     console.log('GJShell Inside (^~^)' +ver);
9542     notices += 1;
9543     if( 2 <= notices ){
9544         document.removeEventListener('mousemove',noticeGShellInside);
9545     }
9546 }
9547 document.addEventListener('mousemove',noticeGShellInside);
9548 noticeGShellInside();

```

```

9549
9550 const FooterName = 'GshFooter'
9551 function DestroyFooter(){
9552     if( (footer = document.getElementById(FooterName)) != null ){
9553         //footer.parentNode.removeChild(footer);
9554         empty = document.createElement('div');
9555         empty.id = 'GshFooter0';
9556         footer.parentNode.replaceChild(empty,footer);
9557     }
9558 }
9559
9560 footer = document.createElement('div');
9561 footer.id = FooterName;
9562 footer.style.backgroundImage = "url("+ITSmoreQR+")";
9563 //GshFooter0.parentNode.appendChild(footer);
9564 GshFooter0.parentNode.replaceChild(footer,GshFooter0);
9565 </script>
9566 <!-- } -->
9567
9568 <!--
9569     border:20px inset #888;
9570 -->
9571
9572 <<span id="WirtualDesktopCodeSpan">
9573 /*
9574 <details><summary>Wirtual Desktop</summary>
9575 <!-- ----- Web Wirtual Desktop // 2020-0927 SatoxITS { -->
9576 <style>
9577 .WirtualSpace {
9578     z-index:0;
9579     xwidth:1280px !important; xheight:720px !important;
9580     width:5120px; height:2880px;
9581     border-width:0px;
9582     xxbackground-color:rgba(32,32,160,0.8);
9583     xxbackground-image:url("WD-WallPaper03.png");
9584     xxbackground-size:100% 100%;
9585     color:#22a;xfont-family:Georgia;font-size:10pt;
9586     xxoverflow:scroll;
9587 }
9588 .WirtualGrid {
9589     z-index:0;
9590     position:absolute;
9591     width:800px; height:500px;
9592     border:1px inset #fff;
9593     color:rgba(192,255,192,0.8);
9594     font-family:Georgia, Courier New;
9595     text-align:right;
9596     vertical-align:middle;
9597     font-size:200px;
9598     text-shadow:4px 4px #ccf;
9599 }
9600 .WD_GridScroll {
9601     z-index:100000;
9602     background-color:rgba(200,200,200,0.1);
9603 }
9604 .WirtualDesktop {
9605     z-index:0;
9606     position:relative;
9607     resize:both !important;
9608     overflow:scroll;
9609     display:block;
9610     min-width:120px !important; min-height:60px !important;
9611     width:800px;
9612     height:500px;
9613     border:10px inset #228;
9614     border-width:30px; border-radius:20px;
9615     background-image:url("WD-WallPaper03.png");
9616     background-size:100% 100%;
9617     color:#22a;font-family:Georgia;font-size:10pt;
9618 }
9619 .comment {
9620     // overflow=scroll seems to bound childrens' view in the element span
9621     // specifying overflow seems fix the position of the element
9622 }
9623 .WirtualBrowserSpan {
9624     z-index:10;
9625     xxxborder:0.5px dashed #fff !important;
9626     border-color:rgba(255,255,255,0.5) !important;
9627     position:relative;
9628     left:100px;
9629     top:100px;
9630     display:block;
9631     resize:both !important;
9632     width:540px;
9633     height:320px;
9634     min-width:40px !important; min-height:20px !important;
9635     max-width:5120px !important; max-height:2880px !important;
9636     background-color:rgba(255,200,255,0.1);
9637     xoverflow:scroll;
9638 }
9639 .xWirtualBrowserLocationBar:focus {
9640     color:#f00;
9641     background-color:rgba(255,128,128,0.2);
9642 }
9643 .xWirtualBrowserLocationBar:active {
9644     color:#f00;
9645     background-color:rgba(128,255,128,0.2);
9646 }
9647 a.WirtualBrowserLocation {
9648     color:#ccc !important;
9649     text-decoration:none !important;
9650 }
9651 a.WirtualBrowserLocation:hover {
9652     color:#fff !important;
9653     text-decoration:underline;
9654 }
9655 .WirtualBrowserLocationBar {
9656     position:absolute;
9657     z-index:100000;
9658     display:block;
9659     width:400px;
9660     height:20px;
9661     padding-left:2px;
9662     line-height:1.1;
9663     vertical-align:middle;
9664     font-size:14px;
9665     color:#fff;
9666     background-color:rgba(128,128,128,0.2);
9667     font-family:Georgia;
9668 }
9669 .WirtualBrowserCommandBar {
9670     position:absolute;
9671     z-index:200000;
9672     xxxdisplay:inline;

```

```

9673     display:block;
9674     width:60px;
9675     height:20px;
9676     line-height:1.1;
9677     vertical-align:middle;
9678     font-size:14px;
9679     color:#fe4;
9680     background-color:rgba(128,128,128,0.1);
9681     font-family:Georgia;
9682     text-align:left;
9683     left:404px;
9684 }
9685 .VirtualBrowserFrame {
9686     xxposition:relative;
9687     position:absolute;
9688     xdisplay:inline;
9689     display:block;
9690     z-index:10;
9691     resize:both !important;
9692     width:480px; height:240px;
9693     min-width:60px; min-height:30px;
9694     max-width:5120px; max-height:2880px;
9695     border-radius:6px;
9696     background-color:rgba(255,255,255,0.9);
9697     border-top:20px solid;
9698     border-right:4px solid;
9699     border-bottom:10px solid;
9700 }
9701 .WinFavicon {
9702     width:16px;
9703     height:16px;
9704     margin:1px;
9705     margin-right:3px;
9706     vertical-align:middle;
9707     background-color:rgba(255,255,255,1.0);
9708 }
9709 .VirtualDesktopMenuBar {
9710     xposition:absolute;
9711     color:#ff;
9712     font-size:7pt;
9713     text-align:right;
9714     padding-right:4px;
9715     background-color:rgba(128,128,128,0.7);
9716 }
9717 .VirtualDesktopCalender {
9718     color:#fff;
9719     font-size:22pt;
9720     text-align:right;
9721     padding-right:4px;
9722     xbackground-color:rgba(255,255,255,0.2);
9723 }
9724 .xxxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9725     display:inline !important; font-size:10pt !important; padding:1px !important;
9726 }
9727 .WD_Config {
9728     display:inline !important;
9729     padding:2px !important;
9730     font-size:10pt !important;
9731     width:60pt !important;
9732     height:12pt !important;
9733     line-height:1.0pt !important;
9734     height:15pt !important;
9735 }
9736 .WD_Button {
9737     display:inline !important;
9738     padding:2px !important;
9739     color:#fff !important;
9740     background-color:#228 !important;
9741     font-size:10pt !important;
9742     width:60pt !important;
9743     height:12pt !important;
9744     line-height:1.0pt !important;
9745     height:16pt !important;
9746     border:2px inset #44a !important;
9747 }
9748 .WD_Href {
9749     display:inline !important;
9750     padding:2px !important;
9751     font-size:9pt !important;
9752     width:120pt !important;
9753     height:12pt !important;
9754     line-height:1.0pt !important;
9755     height:15pt !important;
9756 }
9757
9758 .LiveHtmlCodeviewText {
9759     font-size:10pt;
9760     font-family:Courier New;
9761     xwhite-space:pre;
9762 }
9763
9764 .WD_Panel {
9765     x-index:100 !important;
9766     color:#000 !important;
9767     margin-left:25px !important;
9768     width:800px !important;
9769     padding:4px !important;
9770     border:1px solid #888 !important;
9771     border-radius:6px !important;
9772     background-color:rgba(220,220,220,0.9) !important;
9773     font-size:9pt;
9774     font-family:Courier New;
9775 }
9776 .WD_Help {
9777     font-size:10pt !important;
9778     font-family:Courier New;
9779     line-height:1.2 !important;
9780     color:#000 !important;
9781     width:100% !important;
9782     background-color:rgba(240,240,255,0.8) !important;
9783 }
9784
9785 .WB_Zoom {
9786 }
9787 </style>
9788 <h2>CosmoScreen 0.0.8</h2>
9789 <menu>
9790 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
9791 g ... grid on/off<br>
9792 i ... zoom in<br>
9793 o ... zoom out<br>
9794 s ... save current scope<br>
9795 r ... restore saved scope<br>
9796 </span>

```

```

9797 </menu>
9798 <div class="WD_Panel" draggable="true">
9799 <p><!-- should be on the frame of the WD -->
9800 Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
9801 < <input id="WD_Width_1" class="WD_Config" type="text">
9802 x <input id="WD_Height_1" class="WD_Config" type="text">
9803 wall-paper: <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaper03.png</a>
9804 </p>
9805 <p>
9806 Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
9807 < <input id="WS_1_Width" class="WD_Config" type="text">
9808 x <input id="WS_1_Height" class="WD_Config" type="text">
9809 </p>
9810 <p>
9811 Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
9812 < <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
9813 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
9814 < <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
9815 < <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
9816 </p>
9817 <p>
9818 Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
9819 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
9820 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
9821 shift+wheel for horizontal scroll
9822 </p>
9823 <p>
9824 Scopexx <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
9825 < <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
9826 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
9827 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
9828 </p>
9829 <p>
9830 Scopeyy <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
9831 < <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
9832 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
9833 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
9834 </p>
9835 <p>
9836 Scopezz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
9837 < <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
9838 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
9839 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
9840 </p>
9841 <p>
9842 Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
9843 </p>
9844 <p>
9845 Overflo <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
9846 "scroll" imprisons windows inside the display
9847 </p>
9848 </div>
9849
9850 <div id="WirtualDesktop_1" class="WirtualDesktop" draggable="true" contenteditable="true" style="">
9851 <div id="WirtualDesktop_1_MenuBar" class="WirtualDesktopMenuBar" spellcheck="false">
9852 <i>CosmosScreen 0.0.8</i><span id="WirtualDesktop_1_Clock"></span>
9853 </div>
9854 <div id="WirtualDesktop_1_Calender" class="WirtualDesktopCalender ">00:00</div>
9855 <div align="right"><h1>WirtualSpace 1.</h1></div>
9856 <div id="WirtualDesktop_1_Content" class="WirtualSpace">
9857
9858 <div id="WirtualBrowser_1" class="WirtualBrowserSpan" spellcheck="false" draggable="true">
9859 <div id="WirtualBrowser_1_Location" class="WirtualBrowserLocationBar"></div>
9860 <span id="WirtualBrowser_1_Command" class="WirtualBrowserCommandBar">Reload</span>
9861 <iframe id="WirtualBrowser_1_Frame" class="WirtualBrowserFrame" style=""></iframe>
9862 </div>
9863
9864 <div id="WirtualBrowser_2" class="WirtualBrowserSpan" spellcheck="false" draggable="true">
9865 <div id="WirtualBrowser_2_Location" class="WirtualBrowserLocationBar"></div>
9866 <span id="WirtualBrowser_2_Command" class="WirtualBrowserCommandBar">Reload</span>
9867 <iframe id="WirtualBrowser_2_Frame" class="WirtualBrowserFrame" style=""></iframe>
9868 </div>
9869
9870 <div id="WirtualBrowser_3" class="WirtualBrowserSpan" spellcheck="false" draggable="true">
9871 <div id="WirtualBrowser_3_Location" class="WirtualBrowserLocationBar"></div>
9872 <span id="WirtualBrowser_3_Command" class="WirtualBrowserCommandBar">Reload</span>
9873 <iframe id="WirtualBrowser_3_Frame" class="WirtualBrowserFrame" style=""></iframe>
9874 </div>
9875
9876 <div id="WirtualDesktop_1_GridPlane" class="WirtualSpace"></div>
9877 </div>
9878 </div>
9879
9880 <input class="HtmlCodeviewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">
9881 <span id="WirtualDesktopCodeview"></span>
9882 <script id="WirtualDesktopScript">
9883 function vd_showHtmlCode(){
9884     codespan = document.getElementById('WirtualDesktopCodeSpan');
9885     showHtmlCode(WirtualDesktopCodeview,codespan);
9886     WirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
9887 }
9888 DestroyEventSharingCodeview = function(){
9889     WirtualDesktopCodeview.innerHTML = "";
9890 }
9891
9892 function wdlog(log){
9893     if( GJ_Channel != null ){
9894         GJ_SendMessage('WD '+log);
9895     }
9896     console.log(log);
9897 }
9898 var topMostWin = 10000;
9899 function onEnterWin(e){
9900     t = e.target;
9901     oindex = t.style.zIndex;
9902     //if( oindex == '' ) oindex = 0;
9903     //t.saved_zIndex = oindex;
9904     //t.style.zIndex = 10000;
9905     topMostWin += 1;
9906     t.style.zIndex = topMostWin;
9907     nindex = t.style.zIndex;
9908     wdlog('Enter '+t+' #' + t.id + '(' + oindex + '-' + nindex + ')');
9909     e.stopPropagation();
9910     e.preventDefault();
9911 }
9912 function onClickWin(e){ // can detect click on the thick border? t = e.target;
9913     oindex = t.style.zIndex;
9914     topMostWin += 1;
9915     t.style.zIndex = topMostWin;
9916     nindex = t.style.zIndex;
9917     wdlog('Click '+t+' #' + t.id + '(' + oindex + '-' + nindex + ')');
9918     //e.stopPropagation();
9919     //e.preventDefault();
9920 }

```



```

9921 function onLeaveWin(e){
9922     t = e.target;
9923     //oindex = t.style.zIndex;
9924     //nindex = t.saved_zIndex;
9925     //t.style.zIndex = nindex;
9926     //wdlog('Leave '+e.target+' #' +e.target.id+'('+oindex+'->'+nindex+')');
9927     e.stopPropagation();
9928     e.preventDefault();
9929 }
9930
9931 var WinDragstartX; // event
9932 var WinDragstartY;
9933 var WinDragstartTX; // target
9934 var WinDragstartTY;
9935
9936 function onWinDragstart(e){
9937     WinDragstartX = e.x;
9938     WinDragstartY = e.y;
9939
9940     t = e.target;
9941
9942     //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
9943     //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
9944     if( t.style.left == '' ){
9945         WinDragstartTX = x0 = 0;
9946         t.style.left = '0px';
9947     }else{
9948         //WinDragstartTX = x0 = Number(t.style.left);
9949         WinDragstartTX = x0 = parseInt(t.style.left);
9950     }
9951     if( t.style.top == '' ){
9952         WinDragstartTY = y0 = 0;
9953         t.style.top = '0px';
9954     }else{
9955         //WinDragstartTY = y0 = Number(t.style.top);
9956         WinDragstartTY = y0 = parseInt(t.style.top);
9957     }
9958     if( true ){ // to be undo
9959         t.wasAtX = WinDragstartTX;
9960         t.wasAtY = WinDragstartTY;
9961     }
9962     wdlog('DragSTA #' +t.id
9963         + ' event('+e.x+', '+e.y+')'
9964         + ' position= ' + t.style.position
9965         + ' style left,top('+t.style.left+', '+t.style.top+')'
9966     );
9967     e.stopPropagation();
9968     //e.preventDefault();
9969     return true;
9970 }
9971 function onWinDragEvent(wh,e,set,dolog){
9972     t = e.target;
9973     dx = e.x - WinDragstartX;
9974     dy = e.y - WinDragstartY;
9975     nx = WinDragstartTX + dx;
9976     ny = WinDragstartTY + dy;
9977     log = 'Drag'+wh+' #' +t.id
9978         + ' event0('+WinDragstartX+', '+WinDragstartY+')'
9979         + ' event('+e.x+', '+e.y+')'
9980         + ' diff('+dx+', '+dy+')'
9981         + ' (' + nx + ', ' + ny + ' )'
9982         + ' (' + t.style.left + ', ' + t.style.top + ' )'
9983         + ' wasAt('+ t.wasAtX + ', ' + t.wasAtY + ' )'
9984     ;
9985     if( e.x != 0 || e.y != 0 ){
9986         if( set == true ){
9987             //t.style.x = nx + 'px'; // not effective
9988             //t.style.y = ny + 'px'; // not effective
9989             t.style.left = nx + 'px';
9990             t.style.top = ny + 'px';
9991             log += ' Set';
9992         }else{
9993             log += ' NotSet';
9994             if( !dolog ){
9995                 log = '';
9996             }
9997         }
9998     }else{
9999         log += ' What?'; // the type is event start?
10000         if( !dolog ){
10001             log = '';
10002         }
10003     }
10004     if( and(dolog, log != '' )){
10005         wdlog(log);
10006     }
10007     if( true ){
10008         // should be propagated to parent in FireFox ?
10009         e.stopPropagation();
10010     }
10011     e.preventDefault();
10012     return false;
10013 }
10014 function onWinDrag(e){
10015     return onWinDragEvent('Ing',e,true,false);
10016 }
10017 function onWinDragend(e){
10018     return onWinDragEvent('End',e,false,true);
10019 }
10020 function onWinDragexit(e){
10021     return onWinDragEvent('Exit',e,false,true);
10022 }
10023 function onWinDragover(e){
10024     return onWinDragEvent('Over',e,false,true);
10025 }
10026 function onWinDragenter(e){
10027     return onWinDragEvent('Enter',e,false,true);
10028 }
10029 function onWinDragleave(e){
10030     return onWinDragEvent('Leave',e,false,true);
10031 }
10032 function onWinDragdrop(e){
10033     return onWinDragEvent('Drop',e,false,true);
10034 }
10035 function onFaviconChange(e){
10036     wdlog('--Favicon #' +e.target.id+' href='+e.details.href);
10037 }
10038 var savedSuppressGJShell = false;
10039 function stopGJShell(e){
10040     //wdlog('enter Gsh STOP');
10041     savedSuppressGJShell = SuppressGJShell;
10042     SuppressGJShell = true;
10043     e.stopPropagation();
10044     e.preventDefault();

```

```

10045}
10046function contGShell(e){
10047 //wlog('leave Gsh STOP');
10048 SuppressGShell = savedSuppressGJShell;
10049 e.stopPropagation();
10050 e.preventDefault();
10051}
10052
10053function WD_onKeydown(e){
10054 keycode = e.code;
10055 console.log('Keydown #' + e.target.id + ' +keycode);
10056 if( keycode == 'KeyG' ){
10057   WD_setGrid1(WD_Grid_1);
10058 }else
10059 if( keycode == 'KeyI' ){
10060   WD_doZoomIN();
10061 }else
10062 if( keycode == 'KeyO' ){
10063   WD_doZoomOUT();
10064 }else
10065 if( keycode == 'KeyR' ){
10066   WD_RestoreScope(null);
10067 }else
10068 if( keycode == 'KeyS' ){
10069   WD_SaveScope(null);
10070 }
10071 e.stopPropagation();
10072 e.preventDefault();
10073}
10074function WD_onKeyUp(e){
10075 e.stopPropagation();
10076 e.preventDefault();
10077}
10078VirtualDesktop_1.addEventListener('keydown', e => { WD_onKeydown(e); });
10079VirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onKeydown(e); });
10080WD_Help_1.addEventListener('keydown', e => { WD_onKeydown(e); });
10081WD_Help_1.addEventListener('keyup', e => { WD_onKeyUp(e); });
10082
10083function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
10084 function VirtualBrowserCommand(e,s,l,cmd,F){
10085   command = cmd.innerHTML
10086   if( command == "Reload" ){
10087     href_id = e.target.href_id;
10088     d = document.getElementById(href_id);
10089     wlog('href_tag=#'+href_id+'\n elem=#'+href_id+'\n href='+d);
10090     url = d.innerHTML;
10091     wlog('---- Load href_tag=#'+href_id+'\n elem=#'+href_id+'\n href='+d
10092         +'\n url='+url);
10093     wlog('---- Load target #' + f.id + ' with url=' + url;
10094     f.src = url;
10095   }else{
10096     alert('unknown command'+command+' '+e.target.id+' '+l.id+' '+f.id);
10097   }
10098 }
10099 function onKeyDown(e){
10100   if( e.code == 'Enter' ){
10101     e.stopPropagation();
10102     e.preventDefault();
10103   }
10104 }
10105 function onKeyUp(e){
10106   if( e.code == 'Enter' ){
10107     e.stopPropagation();
10108     e.preventDefault();
10109     // should reload immediately ?
10110   }
10111 }
10112
10113 if( false ){
10114   wlog('start settle VirtualBrowser url='+u +'\n'
10115       + 'id=' + s.id + '\n'
10116       + 'width=' + s.style.width + '\n'
10117       + 'height=' + s.style.height
10118   );
10119 }
10120 // very important for WordPress ??
10121 s.style.width = f.style.width = 501; // for WordPress ...??
10122 s.style.height = f.style.height = 271; // for WordPress ...??
10123 if( false ){
10124   wlog('midway settle VirtualBrowser url='+u +'\n'
10125       + 'id=' + s.id + '\n'
10126       + 'width=' + s.style.width + '\n'
10127       + 'height=' + s.style.height
10128   );
10129 }
10130 s.width = 502; // for WordPress ...??
10131 s.height = 272; // for WordPress ...??
10132 if( false ){
10133   wlog('midway-2 settle VirtualBrowser url='+u +'\n'
10134       + 'id=' + s.id + '\n'
10135       + 'span-width=' + s.width + '\n'
10136       + 'span-height=' + s.height
10137   );
10138 }
10139
10140 s.style.width = w + 'px';
10141 s.style.height = h + 'px';
10142 f.style.width = w + 'px';
10143 f.style.height = h + 'px';
10144 //f.style.setProperty('-webkit-transform','scale('+scale+')');
10145 f.style.setProperty('transform','scale('+scale+')');
10146
10147 //wlog('--x1- u='+u+' width s='+s.style.width+',f='+f.style.width);
10148 //wlog('--x2- u='+u+' width s='+s.style.width+',f='+f.style.width);
10149 s.setAttribute('draggable','true')
10150 f.setAttribute('draggable','false'); // why necessary?
10151 l.setAttribute('draggable','false'); // why necessary?
10152 cmd.setAttribute('draggable','false'); // why necessary?
10153 s.addEventListener('dragstart', e => { onWinDragstart(e); });
10154 s.addEventListener('drag', e => { onWinDrag(e); });
10155 s.addEventListener('exit', e => { onWinDragexit(e); });
10156 s.addEventListener('dragend', e => { onWinDragend(e); });
10157 s.addEventListener('dragexit', e => { onWinDragexit(e); });
10158 s.addEventListener('dragenter', e => { onWinDragenter(e); });
10159 s.addEventListener('dragover', e => { onWinDragover(e); });
10160 s.addEventListener('dragleave', e => { onWinDragleave(e); });
10161 s.addEventListener('drop', e => { onWinDragdrop(e); });
10162
10163 s.addEventListener('mouseenter',e => { onEnterWin(e); });
10164 s.addEventListener('mouseleave',e => { onLeaveWin(e); });
10165
10166 if( false ){
10167   s.style.position = "absolute";
10168   s.style.x = x+'px';

```

```

10169     s.style.left = x+'px';
10170     s.style.y = y+'px';
10171     s.style.top = y+'px';
10172 }else{
10173     s.style.setProperty('position','absolute','important');
10174     s.style.setProperty('x',x+'px','important');
10175     s.style.setProperty('left',x+'px','important');
10176     s.style.setProperty('y',y+'px','important');
10177     s.style.setProperty('top',y+'px','important');
10178 }
10179
10180 favicon = './favicon.ico';
10181 uv1 = u.split('/');
10182 if( 2 <= uv1.length){
10183     uv2 = uv1[1].split('/');
10184     if( 2 <= uv2.length){
10185         if( uv1[0] == 'file' ){
10186             //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/')
10187             // + '/favicon.ico';
10188             favicon = './favicon.ico';
10189         }else{
10190             favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
10191         }
10192     }
10193 }
10194 //wlog("----- favicon-url="+favicon);
10195 href_id = l.id + ' href';
10196 l.innerHTML = ''
10197     + '<a id="'+href_id+'" class="WirtualBrowserLocation" href="'+u+'>'+u+'</a>';
10198 //l.addEventListener('click', e => { onClickWin(e); });
10199 l.addEventListener('mouseenter',e => { stopGShell(e); });
10200 l.addEventListener('mouseleave',e => { contGShell(e); });
10201 l.addEventListener('keydown', e => { onKeyDown(e); });
10202 l.addEventListener('keyup', e => { onKeyUp(e); });
10203
10204 cmd.href_id = href_id;
10205 wlog(' (0)cmd=#'+cmd.id);
10206 wlog(' (1)href_id=#'+href_id);
10207 wlog(' (2)href_id=#'+cmd.href_id);
10208 cmd.addEventListener('click', e => { WirtualBrowserCommand(e,s,l,cmd,f); });
10209
10210 f.style.borderColor = c;
10211 f.src = u;
10212 //f.addEventListener('mouseenter',e => { onEnterWin(e); });
10213 //f.addEventListener('mouseleave',e => { onLeaveWin(e); });
10214
10215 //s.addEventListener('click', e => { onClickWin(e); });
10216 //f.addEventListener('click', e => { wlog('click wbl'); });
10217 f.addEventListener('mozbrowsericonchange',onFaviconChange);
10218
10219 wlog('done settle WirtualBrowser url='+u +'\n'
10220     + 'id=' + s.id + ' '
10221     + 'width=' + s.style.width + ' '
10222     + 'height=' + s.style.height + ' '
10223     + 'cmd=' + cmd.id
10224 );
10225}
10226
10227dt = WirtualDesktop_1;
10228dt.style.width = "800px";
10229dt.style.height = "500px";
10230dt.addEventListener('dragstart',e => { onWinDragstart(e); });
10231dt.addEventListener('drag', e => { onWinDrag(e); });
10232dt.addEventListener('exit', e => { onWinDragexit(e); });
10233
10234function GRonClick(){
10235    WD_SaveScope(null); // should be push
10236    t = event.target;
10237    x = t.getAttribute('data-leftx');
10238    y = t.getAttribute('data-topy');
10239    zoom = WD_Zoom_1_XY.value;
10240    x += zoom;
10241    y += zoom;
10242    WD_doScrollXY(event,x,y);
10243    //alert('scroll #'+t.id+' x='+x+', y='+y);
10244}
10245function WD_setGrid(e){
10246    t = e.target;
10247    WD_setGrid1(t);
10248}
10249function WD_setGrid1(t){
10250    //ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10251    ds = WirtualDesktop_1_GridPlane;
10252    if( t.value == 'GridOn' ){
10253        for( col = 0; col < 16; col++){
10254            for( row = 0; row < 16; row++){
10255                g1 = document.createElement('span');
10256                g1.setAttribute('class','WirtualGrid');
10257                leftx = col * 800;
10258                topy = row * 500;
10259                gid = col + '.' + row
10260                label = '<'+span '
10261                    + 'id="'+gid+'" '+class="WD_GridScroll" '
10262                    + 'contenteditable="false" onclick="GRonClick()" '
10263                    + 'data-leftx=' + leftx + ' ' + 'data-topy=' + topy + ' '
10264                    + '>'
10265                    + gid + '</span>';
10266                console.log('grid '+label);
10267                g1.innerHTML = label;
10268                g1.position = 'relative';
10269                g1.leftx = leftx;
10270                g1.topy = topy;
10271                g1.style.left = g1.leftx + 'px';
10272                g1.style.top = g1.topy + 'px';
10273                if( col % 2 == row % 2 ){
10274                    g1.style.backgroundColor = 'rgba(255,255,255,0.3)';
10275                }
10276                ds.appendChild(g1);
10277            }
10278        }
10279        t.value = 'GridOff';
10280    }else{
10281        ds.innerHTML = '';
10282        t.value = 'GridOn';
10283    }
10284}
10285WD_Grid_1.addEventListener('click', e => { WD_setGrid(e); });
10286
10287var sav_scrollLeft;
10288var sav_scrollTop;
10289var sav_nscale;
10290function WD_SaveScope(e){
10291    sav_scrollLeft = WD_Left_1.value;
10292    sav_scrollTop = WD_Top_1.value;

```

```

10293 sav_nscale = WD_Zoom_1_XY.value;
10294 //console.log('saved zoom='+sav_oscale+', '+sav_nscale);
10295 }
10296 function WD_RestoreScope(e){
10297   WD_Zoom_1_XY.value = sav_nscale;
10298   WD_doZoom();
10299 }
10300 WD_Left_1.value = sav_scrollLeft;
10301 WD_Top_1.value = sav_scrollTop;
10302 WD_doScroll(null);
10303 }
10304 WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
10305 WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
10306 }
10307 function ignoreEvent(e){
10308   e.stopPropagation();
10309   //e.preventDefault();
10310 }
10311 WD_Width_1.value = dt.style.width;
10312 WD_Width_1.addEventListener('keydown', ignoreEvent);
10313 WD_Width_1.addEventListener('keyup', ignoreEvent);
10314 WD_Height_1.value = dt.style.height;
10315 WD_Height_1.addEventListener('keydown', ignoreEvent);
10316 WD_Height_1.addEventListener('keyup', ignoreEvent);
10317 }
10318 function zoomMag(){
10319   return WD_Zoom_1_MAG.value;
10320 }
10321 WD_Zoom_1_MAG.addEventListener('keydown', ignoreEvent);
10322 WD_Zoom_1_MAG.addEventListener('keyup', ignoreEvent);
10323 }
10324 function escale1(e,oscale,nscale){
10325   e.style.setProperty('transform', 'scale('+nscale+')');
10326   rscale = oscale / nscale;
10327   w = parseInt(e.style.width);
10328   h = parseInt(e.style.height);
10329   w = w * rscale; //(oscale/nscale);
10330   h = h * rscale; //(oscale/nscale);
10331   e.style.width = w + 'px';
10332   e.style.height = h + 'px';
10333 }
10334 function scaleWD(ds,oscale,nscale){
10335   if( true ){
10336     escale1(WirtualBrowser_1,oscale,nscale);
10337     escale1(WirtualBrowser_1_Location,oscale,nscale);
10338     escale1(WirtualBrowser_1_Command,oscale,nscale);
10339     escale1(WirtualBrowser_1_Frame,oscale,nscale);
10340 }
10341     escale1(WirtualBrowser_2,oscale,nscale);
10342     escale1(WirtualBrowser_2_Location,oscale,nscale);
10343     escale1(WirtualBrowser_2_Command,oscale,nscale);
10344     escale1(WirtualBrowser_2_Frame,oscale,nscale);
10345 }
10346     escale1(WirtualBrowser_3,oscale,nscale);
10347     escale1(WirtualBrowser_3_Location,oscale,nscale);
10348     escale1(WirtualBrowser_3_Command,oscale,nscale);
10349     escale1(WirtualBrowser_3_Frame,oscale,nscale);
10350 }
10351 }
10352 function WD_doZoom(){
10353   ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10354   oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
10355   nscale = WD_Zoom_1_XY.value;
10356   ds.style.zoom = nscale;
10357   WD_Zoom_1_XY.value = ds.style.zoom;
10358   WD_Zoom_1_XY.ovalue = ds.style.zoom;
10359   scaleWD(ds,oscale,nscale);
10360 }
10361 WD_Zoom_1.addEventListener('click', WD_doZoom);
10362 WD_Zoom_1_XY.addEventListener('keydown', ignoreEvent);
10363 WD_Zoom_1_XY.addEventListener('keyup', ignoreEvent);
10364 }
10365 function WD_doZoomOUT(){
10366   ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10367   oscale = WD_Zoom_1_XY.value;
10368   if( oscale == 0 || oscale == '' ){
10369     oscale = 1;
10370 }
10371     nscale = oscale / zoomMag();
10372     ds.style.zoom = nscale;
10373     WD_Zoom_1_XY.value = ds.style.zoom;
10374     WD_Zoom_1_XY.ovalue = ds.style.zoom;
10375     scaleWD(ds,oscale,nscale);
10376 }
10377 function WD_doZoomIN(){
10378   ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10379   oscale = WD_Zoom_1_XY.value;
10380   if( oscale == 0 || oscale == '' ){
10381     oscale = 1;
10382 }
10383     nscale = oscale * zoomMag();
10384     if( 4 < nscale ){
10385       alert('maybe too large, zoom='+nscale);
10386       return;
10387 }
10388     ds.style.zoom = nscale;
10389     WD_Zoom_1_XY.value = ds.style.zoom;
10390     WD_Zoom_1_XY.ovalue = ds.style.zoom;
10391     scaleWD(ds,oscale,nscale);
10392 }
10393 WD_Zoom_1_OUT.addEventListener('click', WD_doZoomOUT);
10394 WD_Zoom_1_IN.addEventListener('click', WD_doZoomIN);
10395 }
10396 function WD_doResize(e){
10397   dt = WirtualDesktop_1;
10398   dt.style.width = WD_Width_1.value;
10399   dt.style.height = WD_Height_1.value;
10400   WD_Width_1.value = dt.style.width;
10401   WD_Height_1.value = dt.style.height;
10402 }
10403 WD_Resize_1.addEventListener('click', e => { WD_doResize(e); });
10404 }
10405 }
10406 function WD_dorSResize(e){
10407   //alert('Resize Space');
10408   ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10409   ds.style.width = WS_1_Width.value;
10410   ds.style.height = WS_1_Height.value;
10411   WS_1_Width.value = ds.style.width;
10412   WS_1_Height.value = ds.style.height;
10413 }
10414 ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10415 ds.style.width = '5120px';
10416 ds.style.height = '2880px';

```

```

10417WS_1_Width.value = ds.style.width;
10418WS_1_Height.value = ds.style.height;
10419WS_1_Width.addEventListener('keydown',ignoreEvent);
10420WS_1_Width.addEventListener('keyup',ignoreEvent);
10421WS_1_Height.addEventListener('keydown',ignoreEvent);
10422WS_1_Height.addEventListener('keyup',ignoreEvent);
10423WS_Resize_1.addEventListener('click',e => { WD_doRSResize(e); });
10424
10425function WD_doScrollXY(e,sleft,stop){
10426   dt = WirtualDesktop_1;
10427   dt.scrollLeft = sleft;
10428   dt.scrollTop = stop;
10429   WD_Left_1.value = dt.scrollLeft;
10430   WD_Top_1.value = dt.scrollTop;
10431   console.log('--Scroll #' +dt.id+ ' ('+sleft+', '+stop+')');
10432}
10433function WD_doScroll(e){
10434   //dt = WirtualDesktop_1_Content;
10435   dt = WirtualDesktop_1;
10436   sleft = parseInt(WD_Left_1.value);
10437   stop = parseInt(WD_Top_1.value);
10438   dt.scrollLeft = sleft;
10439   dt.scrollTop = stop;
10440   WD_Left_1.value = dt.scrollLeft;
10441   WD_Top_1.value = dt.scrollTop;
10442   console.log('--Scroll #' +dt.id+ ' ('+sleft+', '+stop+')');
10443}
10444WD_Scroll_1.addEventListener('click',e => { WD_doScroll(e); });
10445WD_Left_1.addEventListener('keydown',ignoreEvent);
10446WD_Left_1.addEventListener('keyup',ignoreEvent);
10447WD_Top_1.addEventListener('keydown',ignoreEvent);
10448WD_Top_1.addEventListener('keyup',ignoreEvent);
10449
10450function showScrollPosition(){
10451   if( false )
10452     console.log(
10453       'wstop=' + WirtualDesktop_1.style.top + ',' +
10454       'wsx=' + WirtualDesktop_1.style.y + ',' +
10455       'wsy=' + WirtualDesktop_1.scrollTop + ',' +
10456       'wdtop=' + WirtualDesktop_1_Content.style.top + ',' +
10457       'wdx=' + WirtualDesktop_1_Content.style.y + ',' +
10458       'wds=' + WirtualDesktop_1_Content.scrollTop + ','
10459     );
10460   WD_Left_1.value = WirtualDesktop_1.scrollLeft;
10461   WD_Top_1.value = WirtualDesktop_1.scrollTop;
10462}
10463WirtualDesktop_1.addEventListener('scroll',showScrollPosition);
10464WirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
10465
10466
10467if( false ){
10468w = 1000 + 'px';
10469dt.style.width = w;
10470dt.style.height = "300px";
10471dt.style.resize = "both";
10472dt.style.borderWidth = 50 + 'px';
10473dt.style.borderRadius = 25 + 'px';
10474console.log('--2----- #' +dt.id+ ' style=' +dt.style);
10475console.log('----- #' +dt.id+ ' width=' +dt.style.width);
10476console.log('----- #' +dt.id+ ' left=' +dt.style.left);
10477console.log('----- #' +dt.id+ ' border=' +dt.style.border);
10478}
10479
10480function onDTResize(e){
10481   dt = e.target;
10482   h = parseInt(dt.style.height);
10483   dt.style.borderWidth = (h * 0.075) + 'px';
10484   console.log('----- borderWidgh=' +dt.style.borderWidth);
10485}
10486WirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
10487//console.log('----- #' +dt.id+ ' borderWidth=' +dt.style.getProperty('border-width'));
10488
10489settleWin(
10490   WirtualBrowser_1,
10491   WirtualBrowser_1_Location,
10492   WirtualBrowser_1_Command,
10493   WirtualBrowser_1_Frame,
10494   document.URL,
10495   500,280,50,20,'#262',1.0);
10496settleWin(
10497   WirtualBrowser_2,
10498   WirtualBrowser_2_Location,
10499   WirtualBrowser_2_Command,
10500   WirtualBrowser_2_Frame,
10501   'https://its-more.jp/ja_jp/',
10502   500,280,150,100,'#448',1.0);
10503
10504settleWin(
10505   WirtualBrowser_3,
10506   WirtualBrowser_3_Location,
10507   WirtualBrowser_3_Command,
10508   WirtualBrowser_3_Frame,
10509   //'../gshell/gsh.go.html',
10510   //'http://gshell.org/gshell/gsh.go.html',
10511   'https://golang.org',
10512   500,280,250,180,'#444',1.0);
10513   //1000,720,0,0,'#444',0.125);
10514   //1200,720,-100,-50,'#444',0.4);
10515
10516function WD_ClockUpdate(e){
10517   WirtualDesktop_1_Clock.innerHTML = DateShort();
10518   WirtualDesktop_1_Calender.innerHTML = DateHourMin();
10519}
10520window.setInterval(WD_ClockUpdate,500);
10521//GJ_Join();
10522
10523Destroy_WirtualDesktop = function(){
10524   WirtualDesktop_1.style = "";
10525
10526   WirtualBrowser_1.removeAttribute('style');
10527   WirtualBrowser_1_Location.innerHTML = '';
10528   WirtualBrowser_1_Frame.removeAttribute('src');
10529   WirtualBrowser_1_Frame.removeAttribute('style');
10530   WirtualBrowser_1_Frame.style="";
10531
10532   WirtualBrowser_2.removeAttribute('style');
10533   WirtualBrowser_2_Location.innerHTML = '';
10534   WirtualBrowser_2_Frame.removeAttribute('src');
10535   WirtualBrowser_2_Frame.style="";
10536
10537   WirtualBrowser_3.removeAttribute('style');
10538   WirtualBrowser_3_Location.innerHTML = '';
10539   WirtualBrowser_3_Frame.removeAttribute('src');
10540   WirtualBrowser_3_Frame.style="";

```

```

10541
10542   GJFactory_1.style = "";
10543   iframe0.style = "";
10544   WirtualDesktop_1.style = "";
10545 }
10546
10547</script>
10548<!-- WirtualDesktop } -->
10549</details>
10550*/ </span>
10551
10552<!-- Work { ----- -->
10553</span id="SBSidebar_WorkCodeSpan">
10554/*
10555<details><summary>SBSidebar</summary>
10556<!-- SBSidebar // 2020-0928 SatoxITS { -->
10557<h2>SBSidebar</h2>
10558<input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
10559<input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
10560<input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
10561<span id="SBSidebar_WorkCodeView"></span>
10562<script id="SBSidebar_WorkScript">
10563function SBSidebar_openWorkCodeView(){
10564   function SBSidebar_showWorkCode(){
10565       showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
10566   }
10567   SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
10568}
10569SBSidebar_openWorkCodeView(); // should be invoked by an event
10570
10571console.log('-- SbsSlider // 2020-1006-01 SatoxITS --');
10572function SetSidebar(){
10573   sidebar = document.getElementById('secondary');
10574//   console.log('primary='+primary+ ' + secondary='+secondary+ ' + main='+main+ ' ');
10575   wrap = sidebar.parentNode;
10576   console.log('-- SbsSlider parent is '+wrap+', #' + wrap.id + ' .' + wrap.class);
10577//wrap = wrap.parentNode;
10578//console.log('-- SbsSlider parent is '+wwrap+', #' + wwrap.id + ' .' + wwrap.class);
10579//wwrap = wwrap.parentNode;
10580//console.log('-- SbsSlider parent is '+wwwrap+', #' + wwwrap.id + ' .' + wwwrap.class);
10581//nsb = sidebar.cloneNode();
10582   nsb = sidebar;
10583   nsb.style.width = '100%';
10584   slider = document.createElement('div');
10585   slider.id = 'SbSlider';
10586   slider.appendChild(nsb);
10587   slider.setAttribute('class','SbSlider');
10588   nsb.style.position = 'relative';
10589   slider.style.position = 'fixed';
10590   slider.style.display = 'block'; //inline';
10591   slider.style.zIndex = 100000;
10592   // nsb.style.zIndex = 200000;
10593   nsb.style.position = 'absolute';
10594   nsb.style.minWidth = '80px';
10595   nsb.style.left = '0px';
10596   nsb.style.top = '0px';
10597
10598   w = window.innerWidth;
10599   console.log('SliderWidth '+w+' : ',(w/3)+'px');
10600   if( w < 640 ){
10601       slider.style.setProperty('width',(w/3) + 'px','important');
10602   }
10603   main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
10604
10605   slider.style.resize = "both";
10606   slider.draggable = "true";
10607   wrap.appendChild(slider);
10608   console.log('-- added SbsSlider');
10609   //nsb.addEventListener('scroll',SbScrolled);
10610
10611   buttons = document.createElement('div');
10612   buttons.id = 'NaviButtons';
10613   buttons.setAttribute('class','NaviButtons');
10614   buttons.align = "center";
10615   buttons.innerHTML += '<+><p><a href="#TopOfPost" draggable="true">TOP<+></p>';
10616   buttons.innerHTML += '<+><p><a href="#EndOfPost" draggable="true">END<+></p>';
10617   page.appendChild(buttons);
10618   buttons.style.position = 'fixed';
10619   buttons.style.zIndex = 30000;
10620   buttons.style.width = '180%';
10621   buttons.style.top = '320px';
10622   buttons.style.left = parseInt(w) * 1.0 + 'px';
10623   console.log('-- SbsSlider installed (^~)/ SatoxITS');
10624}
10625//window.addEventListener('load',SetSidebar); // after load
10626DestroyNaviButtons = function(){
10627   nb = document.getElementById('NaviButtons');
10628   if( nb != null ){
10629       nb.parentNode.removeChild(NaviButtons);
10630   }
10631}
10632</script>
10633
10634// 2020-1006 its-more.jp-blog-60000-style.css
10635<!-- {
10636<style>
10637#NaviButtons {
10638   position:fixed;
10639   display:block;
10640   xwidth:100%;
10641   xtop:100px;
10642   xxleft:10px;
10643   z-index:30000;
10644   font-size:10pt;
10645   color:#2ff !important;
10646   text-align:center;
10647   background-color:rgba(230,230,230,0.01);
10648}
10649#NaviButtons a {
10650   color:#2a2 !important;
10651   font-size:20px;
10652   text-align:center;
10653   xtext-shadow:2px 2px #8ff;
10654   resize:both;
10655   padding:6px;
10656   margin:10px;
10657   border:1px solid #288 !important;
10658   border-radius:3px;
10659   background-color:rgba(160,160,160,0.05);
10660}
10661#SbSlider {
10662   overflow:auto;
10663   resize:both !important;
10664   xxoverflow-y:hidden !important;

```

