

```

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--0.6.3--2020-10-08--SatoxITS</span>
7 <title id="GshTitle">GShell-0.6.3 by SatoxITS</title>
8
9 <div id="GshTopbar" class="MetaWindow"><div id="GshMetas"></div></div>
10 <div id="GshSidebar" class="SBSidebar"><div id="GshIndexer"></div></div>
11 <div id="GshMain">
12 <img id="ConfigIcon" class="ConfigIcon">
13 <header id="GshBanner" height="100px" onclick="shiftBG();">
14 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.6.3 // 2020-10-08 // SatoxITS</note></div>
15 </header>
16
17 <!-- Work { ----- -->
18 <span id="Topbar_WorkCodeSpan">
19 /*
20 <details><summary>Topbar</summary>
21 <!-- ----- Topbar // 2020-1008 SatoxITS { -->
22 <h2>Topbar</h2>
23 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
24 <input id="Topbar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
25 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
26 <span id="Topbar_WorkCodeView"></span>
27 </details>
28
29 <style>
30 .ConfigIcon {
31   position:absolute;
32   top:-6px;
33   left:92%;
34   width:32px;
35   height:32px;
36 }
37 .MetaWindow {
38   z-index:1000;
39   position:relative;
40   display:block;
41   overflow:visible !important;
42   width:99.9%;
43   height:22px;
44   top:-22px;
45   border:1px solid #22a;
46   margin:0px;
47   left:0.0%;
48   line-height:1.0;
49   font-family:Georgia;
50   color:#fff;
51   font-size:12pt;
52   text-align:center;
53   vertical-align:middle;
54   padding:4px;
55   xbackground-color:rgba(0,8,170,0.8);
56   background-color:#3a4861;xxx-PBlue;
57   vertical-align:middle;
58 }
59 .MetaWindow:hover {
60   color:#000;
61   border:1px solid #22a;
62   background-color:rgba(255,255,255,1.0);
63 }
64 </style>
65 <script>
66 function Topbar_openWorkCodeView(){
67   function Topbar_showWorkCode(){
68     showHtmlCode(Topbar_WorkCodeView,Topbar_WorkCodeSpan);
69   }
70   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
71 }
72 Topbar_openWorkCodeView();
73 function Gshell_initTopbar(){
74   GshTopbar.innerHTML = GshTitle.innerHTML;
75   GshTopbar.appendChild(ConfigIcon);
76   ConfigIcon.src = ConfigICON_DATA;
77 }
78 </script>
79 <!-- Topbar_WorkCodeSpan } -->
80 */ </span>
81 <!-- Work } ----- -->
82
83
84 <!-- Work { ----- -->
85 <span id="Indexer_WorkCodeSpan">
86 /*
87 <details><summary>Indexer</summary>
88 <!-- ----- Indexer // 2020-1007 SatoxITS { -->
89 <h2>Indexer</h2>
90 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
91 <input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
92 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
93 <span id="Indexer_WorkCodeView"></span>
94 </details>
95 <style id="SidebarIndex">
96 #gsh {
97   display:block;
98   xoverflow:scroll !important;
99 }
100 #GshMain {
101   position:relative;
102   width:80% !important;
103   left:19.5% !important;
104 }
105 #GshSidebar {
106   z-index:0;
107   position:relative !important;
108   overflow:auto;
109   resize:both !important;
110   xoverflow-y:hidden !important;
111   height:100px !important;
112   xxxdisplay:inline !important;
113   left:0px;
114   top:0px;
115   width:19.5%;
116   min-width:80px;
117   height:100% !important;
118   color:#f00;
119   xbackground-color:rgba(64,64,64,0.5);
120   xbackground-color:#DFE3EB;xxx-PBlue;
121   background-color:#e0e0e0;xxx-PBlue;
122 }
123 #GshSidebar:hover {
124   z-index:10000000;

```

```

125 overflow-x:visible !important;
126 background-color:rgba(255,255,255,0.7);
127 width:50%;
128 }
129 #GshIndexer {
130 z-index:0;
131 position:relative;
132 resize:both !important;
133 height:100%;
134 left:0px;
135 top:0px;
136 scroll-behavior: overflow !important;
137 padding-left:4pt;
138 font-size:0.5em;
139 white-space:nowrap;
140 xxx-background-color:rgba(64,160,64,0.6) !important;
141 color:#7794c6;xxx-PBlue;
142 xxxbackground-color:#DFE3EB;xxx-PBlue;
143 background-color:#eeeeee;xxx-PBlue;
144 }
145 #GshIndexer:hover {
146 z-index:1000000;
147 overflow-x:visible !important;
148 color:#000000 !important;xxx-PBlue;
149 xxxbackground-color:#FFFFFF;xxx-PBlue;
150 background-color:rgba(255,255,255,0.7);
151 padding-right:0px;
152 width:80%;
153 }
154 #GshIndexer:select {
155 color:#000000 !important;xxx-PBlue;
156 background-color:#FFFFFF;xxx-PBlue;
157 }
158 .IndexLine {
159 font-size:8pt !important;
160 font-family:Georgla;
161 display:block;
162 xxx-color:#fff;
163 xxx-color:#fff;xxx-PBlue;
164 xxx-color:#41516d;xxx-PBlue;
165 xxx-color:#7794c6;xxx-PBlue;
166 padding-right:4pt;
167 }
168 .IndexLine:hover {
169 font-size:10pt!important;
170 xxx-color:#228;
171 xxx-background-color:#fff;
172 xxxcolor:#fff;xxx-PBlue;
173 color:#516487;xxx-PBlue;
174 background-color:rgba(220,220,255,1.0);xxx-PBlue;
175 xxxtext-shadow:1px 1px #3f3;
176 text-shadow:1px 1px #eee;
177 xxxbackground-color:#516487;xxx-PBlue;
178 xxxtext-decoration:underline !important;
179 }
180 </style>
181
182 <script id="Indexer_WorkScript">
183 function Indexer_openWorkCodeView(){
184 function Indexer_showWorkCode(){
185 showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
186 }
187 Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
188 }
189 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
190 Indexer_openWorkCodeView();
191
192 var iserno = 0;
193 var GeneratedId = 0;
194 function generateIndex(ni,e,chv,nch,ht){
195 // https://developer.mozilla.org/en-US/docs/Web/API/Element
196 c = '';
197 if( e.classList != null ){
198 c = e.classList.value;
199 }
200 console.log('-- <'+e.nodeName+'> #' +e.id+' .'+c+' '+e.attributes);
201 if( e.nodeName == '#text' ){ return ''; }
202 if( e.nodeName == '#comment' ){ return ''; }
203 if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
204 id = e.innerHTML;
205 GeneratedId += 1;
206 eid = 'GeneratedId-'+GeneratedId;
207 e.id = eid;
208 }else
209 if( e.nodeName == 'SUMMARY' ){
210 id = e.innerHTML;
211 GeneratedId += 1;
212 eid = 'GeneratedId-'+GeneratedId;
213 e.id = eid;
214 }else
215 if( and(e.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxentry-content' ) ){
216 console.log('-- DIV entry-content begin');
217 id = e.innerHTML;
218 GeneratedId += 1;
219 eid = 'GeneratedId-'+GeneratedId;
220 e.id = eid;
221 console.log('-- DIV entry-content end hash-child='+e.hasChildNodes());
222 }else
223 if( e.id == '' || e.id == 'undefined' ){
224 return '';
225 }else{
226 id = '#' +e.id;
227 eid = e.id;
228 }
229 iserno += 1;
230 ht = '<'+div id="GeneratedEref '+iserno+' " class="IndexLine" href="'+eid+' ">
231 + isernov' +ni+' '+e.nodeName + ' ' + id;
232 if( e.id == '' || e.id == 'undefined' ){ return ht + '<'+div>'; }
233 if( !e.hasChildNodes() ){ return ht + '<'+div>'; }
234 chv = e.childNodes;
235 nch = e.childNodes.length;
236 if( chv != null ){ nch = chv.length; }
237 ht += ' ('+nch+')' + '<'+div>';
238 for( let i = 0; i < chv.length; i++ ){
239 sec = ni+'.'+i;
240 if( ni == '' ){ sec = i; }
241 ht += generateIndex(sec,chv[i],null,0);
242 }
243 return ht;
244 }
245 function onClickIndex(e){
246 tid = e.target.id;
247 tge = document.getElementById(tid);
248 eid = tge.getAttribute('href');

```

```

249 rx = tge.getBoundingClientRect().left.toFixed(0)
250 ry = tge.getBoundingClientRect().top.toFixed(0)
251 if( false ){
252     alert('index clicked mouse(x='+e.x+', y='+e.y+')'
253         + '\ntid=#'+ tid + ' rx='+rx + ',ry='+ry
254         + '\neid=' + eid + '\n'
255         + '\nhtml='+ tge.outerHTML);
256 }
257 ee = document.getElementById(eid);
258 sx = 'NaN';
259 sy = ee.getBoundingClientRect().top;
260 console.log('sx='+sx+',sy='+sy);
261 ee.scrollIntoView()
262 window.scrollTo(sx,sy)
263 //window.scrollTo({left:'Non',top:sy,behavior:'smooth'});
264 }
265 function Indexer_afterLoaded(){
266     sideindex = document.getElementById('GshIndexer');
267     ht = '<'+h3>G-Index<'+/h3>';
268     ht += generateIndex("",document.getElementById('gsh'),null,0,'');
269     if( (pri = document.getElementById('primary')) != null ){
270         ht += generateIndex("",pri,null,0,'');
271     }
272     ht += '<'+br>';
273     ht += '<'+br>';
274     ht += '<'+br>';
275     ht += '<'+br>';
276     sideindex.innerHTML = ht;
277     sideindex.addEventListener('click',onClickIndex);
278 }
279 if( (pri = document.getElementById('primary')) != null ){
280     console.log('-- Seems in WordPress');
281     pri.style.zIndex = 2000;
282 }
283 GshSidebar.style.setProperty('position','relative','important');
284 GshSidebar.style.top = '-1400px';
285 //GshSidebar.style.setProperty('position','absolute','important');
286 //GshSidebar.style.top = '0px';
287 }
288 GshSidebar.style.setProperty('width','200px','important');
289 GshSidebar.style.setProperty('overflow','scroll','important');
290 GshSidebar.style.resize = 'both';
291 }
292 GshSidebar.style.left = '-100px';
293 GshIndexer.style.left = '100px';
294 GshIndexer.style.height = '1400px';
295 gsh.appendChild(GshSidebar); // change parent
296 }else{
297     console.log('-- Seems not in WordPress');
298     GshSidebar.style.setProperty('position','fixed','important');
299 }
300 }
301 //document.addEventListener('load',Indexer_afterLoaded);
302 }
303 DestroyIndexBar = function(){
304     sideindex = document.getElementById('GshIndexer');
305     sideindex.innerHTML = "";
306 }
307 </script>
308
309 <!-- Indexer_WorkCodeSpan -->
310 */ </span>
311 <!-- ----- Work } ----- -->
312
313
314 /*
315 <h2>GShell // a General purpose Shell built on the top of Golang</h2>
316 <p>
317 <note>
318 It is a shell for myself, by myself, of myself. --SatoxITS(^-^ )
319 <a href="gsh-0.6.2.go.html">prev.</a>
320 </note>
321 </p>
322 <div id="GJFactory_x"></div>
323
324 <div>
325 <span id="GshMenu">
326 <span class="GshMenu" id="GshMenuEdit" onclick="html_edit();">Edit</span>
327 <span class="GshMenu" id="GshMenuSave" onclick="html_save();">Save</span>
328 <span class="GshMenu" id="GshMenuLoad" onclick="html_load();">Load</span>
329 <span class="GshMenu" id="GshMenuVers" onclick="html_ver0();">Vers</span>
330 <span id="gsh-WinId" onclick="win_jump('0.1');">0</span>
331 <span class="GshMenu" id="gsh-menu-exit" onclick="html_close();"></span>
332 <span class="GshMenu" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
333 <span class="GshMenu" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
334 <span class="GshMenu" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
335 <span class="GshMenu" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
336 <span class="GshMenu" id="GshMenuSign" onclick="html_sign(this);" style="">Source</span>
337 <!-- | <span id="gsh-menu-pure" onclick="html_pure(this);">Pure</span> -->
338 </span>
339 </div>
340 */
341
342 /*
343 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
344 <h3>Fun to create a shell</h3>
345 <p>For a programmer, it must be far easy and fun to create his own simple shell
346 rightly fitting to his favor and necessities, than learning existing shells with
347 complex full features that he never use.
348 I, as one of programmers, am writing this tiny shell for my own real needs,
349 totally from scratch, with fun.
350 </p><p>
351 For a programmer, it is fun to learn new computer languages. For long years before
352 writing this software, I had been specialized to C and early HTML2 :-).
353 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
354 on demand as a novice of these, with fun.
355 </p><p>
356 This single file "gsh.go", that is executable by Go, contains all of the code written
357 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
358 HTML file that works as the viewer of the code of itself, and as the "home page" of
359 this software.
360 </p><p>
361 Because this HTML file is a Go program, you may run it as a real shell program
362 on your computer.
363 But you must be aware that this program is written under situation like above.
364 Needless to say, there is no warranty for this program in any means.
365 </p>
366 <address>Aug 2020, SatoxITS (sato@its-more.jp)</address>
367 </details>
368 */
369 /*
370 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
371 </p>
372 <h3>Cross-browser communication</h3>

```

```

373 <p>
374 ... to be written ...
375 </p>
376 <h3>Vi compatible command line editor</h3>
377 <p>
378 The command line of GShell can be edited with commands compatible with
379 <a href="https://www.washington.edu/computing/unix/vi.html"><b>vi</b></a>.
380 As in vi, you can enter <i><b>command mode</b></i> by <b>ESC</b> key,
381 then move around in the history by <b>code>j k / ? n N</code></b>,
382 or within the current line by <b>code>l h f w b o $ %</code></b> or so.
383 </p>
384 </details>
385 */
386 /*
387 <details id="gsh-gindex">
388 <summary>Index</summary><div class="gsh-src">
389 Documents
390 <span class="gsh-link" onclick="jumpto_JavaScriptView();">Command summary</span>
391 Go lang part<span class="gsh-link" onclick="document.getElementById('gsh-gocode').open=true;">
392 Package structures
393 <a href="#import">import</a>
394 <a href="#struct">struct</a>
395 Main functions
396 <a href="#comexpansion">str-expansion</a> // macro processor
397 <a href="#finder">finder</a> // builtin find + du
398 <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
399 <a href="#plugin">plugin</a> // plugin commands
400 <a href="#ex-commands">system</a> // external commands
401 <a href="#builtin">builtin</a> // builtin commands
402 <a href="#network">network</a> // socket handler
403 <a href="#remote-sh">remote-sh</a> // remote shell
404 <a href="#redirect">redirect</a> // StdIn/Out redirection
405 <a href="#history">history</a> // command history
406 <a href="#rusage">rusage</a> // resouce usage
407 <a href="#encode">encode</a> // encode / decode
408 <a href="#IME">IME</a> // command line IME
409 <a href="#getline">getline</a> // line editor
410 <a href="#scanf">scanf</a> // string decomposer
411 <a href="#interpreter">interpreter</a> // command interpreter
412 <a href="#main">main</a>
413 </span>
414 JavaScript part
415 <a href="#script-src-view" class="gsh-link" onclick="jumpto_JavaScriptView();">Source</a>
416 <a href="#gsh-data-frame" class="gsh-link" onclick="jumpto_DataView();">Builtin data</a>
417 CSS part
418 <a href="#style-src-view" class="gsh-link" onclick="jumpto_StyleView();">Source</a>
419 References
420 <a href="#" class="gsh-link" onclick="jumpto_WholeView();">Internal</a>
421 <a href="#gsh-reference" class="gsh-link" onclick="jumpto_ReferenceView();">External</a>
422 Whole parts
423 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Source</a>
424 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Download</a>
425 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Dump</a>
426 </div>
427 </details>
428 */
429 </details id="gsh-gocode">
430 <summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
431 // gsh - Go lang based Shell
432 // (c) 2020 ITS more Co., Ltd.
433 // 2020-0807 created by SatoxITS (sato@its-more.jp)
434
435 package main // gsh main
436
437 // <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
438 import (
439     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
440     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
441     "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
442     "sort" // <a href="https://golang.org/pkg/sort/">sort</a>
443     "time" // <a href="https://golang.org/pkg/time/">time</a>
444     "bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
445     "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
446     "os" // <a href="https://golang.org/pkg/os/">os</a>
447     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
448     "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
449     "net" // <a href="https://golang.org/pkg/net/">net</a>
450     "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
451     "html" // <a href="https://golang.org/pkg/html/">html</a>
452     "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
453     "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
454     "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
455     "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
456     "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
457     "gshdata" // gshell's logo and source code
458     "hash/crc32" // <a href="https://golang.org/pkg/unicode/hash/crc32/">crc32</a>
459     "golang.org/x/net/websocket"
460 )
461
462 // // 2020-0906 added,
463 // // <a href="https://golang.org/cmd/cgo/">CGO</a>
464 // #include "poll.h" // <poll.h> to be closed as HTML tag i-p
465 // typedef struct { struct pollfd fdv[8]; } pollFdv;
466 // int pollx(pollFdv *fdv, int nfd, int timeout){
467 //     return poll(fdv->fdv,nfds,timeout);
468 // }
469 //
470 import "C"
471
472 // // 2020-0906 added,
473 func CFPollIn1(fp*os.File, timeoutUs int)(ready uintptr){
474     var fdv = C.pollFdv{}
475     var nfd = 1
476     var timeout = timeoutUs/1000
477
478     fdv.fdv[0].fd = C.int(fp.Fd())
479     fdv.fdv[0].events = C.POLLIN
480     if( 0 < EventRecvFd ){
481         fdv.fdv[1].fd = C.int(EventRecvFd)
482         fdv.fdv[1].events = C.POLLIN
483         nfd += 1
484     }
485     r := C.pollx(&fdv,C.int(nfd),C.int(timeout))
486     if( r <= 0 ){
487         return 0
488     }
489     if (int(fdv.fdv[1].revents) & int(C.POLLIN)) != 0 {
490         //fprintf(stderr,"--De-- got Event\n");
491         return uintptr(EventFdOffset + fdv.fdv[1].fd)
492     }
493     if (int(fdv.fdv[0].revents) & int(C.POLLIN)) != 0 {
494         return uintptr(NormalFdOffset + fdv.fdv[0].fd)
495     }
496     return 0

```

```

497 }
498
499 const (
500     NAME = "gsh"
501     VERSION = "0.6.3"
502     DATE = "2020-10-08"
503     AUTHOR = "SatoxITS(^-^)"
504 )
505 var (
506     GSH_HOME = ".gsh" // under home directory
507     GSH_PORT = 9999
508     MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
509     PROMPT = "> "
510     LINESIZE = (8*1024)
511     PATHSEP = ";" // should be ";" in Windows
512     DIRSEP = "/" // canbe \ in Windows
513 )
514
515 // -xX logging control
516 // --A-- all
517 // --I-- info.
518 // --D-- debug
519 // --T-- time and resource usage
520 // --W-- warning
521 // --E-- error
522 // --F-- fatal error
523 // --Xn- network
524
525 // <a name="struct">Structures</a>
526 type GCommandHistory struct {
527     StartAt time.Time // command line execution started at
528     EndAt time.Time // command line execution ended at
529     ResCode int // exit code of (external command)
530     CmdError error // error string
531     OutData *os.File // output of the command
532     FoundFile []string // output - result of ufind
533     Rusagev [2]syscall.Rusage // Resource consumption, CPU time or so
534     CmdId int // maybe with identified with arguments or impact
535     // redirection commands should not be the CmdId
536     WorkDir string // working directory at start
537     WorkDirX int // index in ChdirHistory
538     CmdLine string // command line
539 }
540 type GChdirHistory struct {
541     Dir string
542     MovedAt time.Time
543     CmdIndex int
544 }
545 type CmdMode struct {
546     Background bool
547 }
548 type Event struct {
549     when time.Time
550     event int
551     evarg int64
552     CmdIndex int
553 }
554 var CmdIndex int
555 var Events []Event
556 type PluginInfo struct {
557     Spec *plugin.Plugin
558     Addr plugin.Symbol
559     Name string // maybe relative
560     Path string // this is in Plugin but hidden
561 }
562 type GServer struct {
563     host string
564     port string
565 }
566
567 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
568 const ( // SumType
569     SUM_ITEMS = 0x000001 // items count
570     SUM_SIZE = 0x000002 // data length (simply added)
571     SUM_SIZEHASH = 0x000004 // data length (hashed sequence)
572     SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
573     // also envelope attributes like time stamp can be a part of digest
574     // hashed value of sizes or mod-date of files will be useful to detect changes
575
576     SUM_WORDS = 0x000010 // word count is a kind of digest
577     SUM_LINES = 0x000020 // line count is a kind of digest
578     SUM_SUM64 = 0x000040 // simple add of bytes, useful for human too
579
580     SUM_SUM32_BITS = 0x000100 // the number of true bits
581     SUM_SUM32_2BYTE = 0x000200 // 16bits words
582     SUM_SUM32_4BYTE = 0x000400 // 32bits words
583     SUM_SUM32_8BYTE = 0x000800 // 64bits words
584
585     SUM_SUM16_BSD = 0x001000 // UNIXsum -sum -bsd
586     SUM_SUM16_SYSV = 0x002000 // UNIXsum -sum -sysv
587     SUM_UNIXPTLE = 0x004000
588     SUM_CRCIEEE = 0x008000
589 )
590 type CheckSum struct {
591     Files int64 // the number of files (or data)
592     Size int64 // content size
593     Words int64 // word count
594     Lines int64 // line count
595     SumType int
596     Sum64 uint64
597     Crc32Table crc32.Table
598     Crc32Val uint32
599     Sum16 int
600     Ctime time.Time
601     Atime time.Time
602     Mtime time.Time
603     Start time.Time
604     Done time.Time
605     RusgAtStart [2]syscall.Rusage
606     RusgAtEnd [2]syscall.Rusage
607 }
608 type ValueStack [][]string
609 type GshContext struct {
610     StartDir string // the current directory at the start
611     GetLine string // gsh-getline command as a input line editor
612     ChdirHistory []GChdirHistory // the 1st entry is wd at the start
613     gshPA syscall.ProcAttr
614     CommandHistory []GCommandHistory
615     CmdCurrent GCommandHistory
616     Background bool
617     BackgroundJobs []int
618     LastRusage syscall.Rusage
619     GshHomeDir string
620     TerminalId int

```

```

621 CmdTrace    bool // should be [map]
622 CmdTime     bool // should be [map]
623 PluginFuncs []PluginInfo
624 iValues     []string
625 idelimiter  string // field separator of print out
626 iFormat     string // default print format (of integer)
627 iValStack   ValueStack
628 LastServer  GServer
629 RSERVER    string // [gsh://]host[:port]
630 RWD        string // remote (target, there) working directory
631 lastChecksum CheckSum
632 }
633
634 func nsleep(ns time.Duration){
635     time.Sleep(ns)
636 }
637 func usleep(ns time.Duration){
638     nsleep(ns*1000)
639 }
640 func msleep(ns time.Duration){
641     nsleep(ns*1000000)
642 }
643 func sleep(ns time.Duration){
644     nsleep(ns*1000000000)
645 }
646
647 func strBegins(str, pat string)(bool){
648     if len(pat) <= len(str){
649         yes := str[0:len(pat)] == pat
650         //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat, yes)
651         return yes
652     }
653     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
654     return false
655 }
656 func isin(what string, list []string) bool {
657     for _, v := range list {
658         if v == what {
659             return true
660         }
661     }
662     return false
663 }
664 func isinX(what string,list[]string)(int){
665     for i,v := range list {
666         if v == what {
667             return i
668         }
669     }
670     return -1
671 }
672
673 func env(opts []string) {
674     env := os.Environ()
675     if isin("-", opts){
676         sort.Slice(env, func(i,j int) bool {
677             return env[i] < env[j]
678         })
679     }
680     for _, v := range env {
681         fmt.Printf("%v\n",v)
682     }
683 }
684
685 // - rewriting should be context dependent
686 // - should postpone until the real point of evaluation
687 // - should rewrite only known notation of symbol
688 func scanInt(str string)(val int, leng int){
689     leng = -1
690     for i,ch := range str {
691         if '0' <= ch && ch <= '9' {
692             leng = i+1
693         }else{
694             break
695         }
696     }
697     if 0 < leng {
698         ival,_ := strconv.Atoi(str[0:leng])
699         return ival,leng
700     }else{
701         return 0,0
702     }
703 }
704 func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
705     if len(str[i+1:]) == 0 {
706         return 0,rstr
707     }
708     hi := 0
709     histlen := len(gshCtx.CommandHistory)
710     if str[i+1] == '|' {
711         hi = histlen - 1
712         leng = 1
713     }else{
714         hi,leng = scanInt(str[i+1:])
715         if leng == 0 {
716             return 0,rstr
717         }
718         if hi < 0 {
719             hi = histlen + hi
720         }
721     }
722     if 0 <= hi && hi < histlen {
723         var ext byte
724         if 1 < len(str[i+leng:]) {
725             ext = str[i+leng:][1]
726         }
727         //fmt.Printf("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
728         if ext == 'f' {
729             leng += 1
730             xlist := []string{}
731             list := gshCtx.CommandHistory[hi].FoundFile
732             for _,v := range list {
733                 //list[i] = escapeWhiteSP(v)
734                 xlist = append(xlist,escapeWhiteSP(v))
735             }
736             //rstr += strings.Join(list," ")
737             rstr += strings.Join(xlist," ")
738         }else
739         if ext == '@' || ext == 'd' {
740             // !N@ .. workdir at the start of the command
741             leng += 1
742             rstr += gshCtx.CommandHistory[hi].WorkDir
743         }else{
744             rstr += gshCtx.CommandHistory[hi].CmdLine

```

```

745     }
746   }else{
747     leng = 0
748   }
749   return leng,rstr
750 }
751 func escapeWhiteSP(str string)(string){
752   if len(str) == 0 {
753     return "\\z" // empty, to be ignored
754   }
755   rstr := ""
756   for _,ch := range str {
757     switch ch {
758       case '\\': rstr += "\\\\"
759       case ' ': rstr += "\\s"
760       case '\t': rstr += "\\t"
761       case '\r': rstr += "\\r"
762       case '\n': rstr += "\\n"
763       default: rstr += string(ch)
764     }
765   }
766   return rstr
767 }
768 func unescapeWhiteSP(str string)(string){ // strip original escapes
769   rstr := ""
770   for i := 0; i < len(str); i++ {
771     ch := str[i]
772     if ch == '\\' {
773       if i+1 < len(str) {
774         switch str[i+1] {
775           case 'z':
776             continue;
777         }
778       }
779       rstr += string(ch)
780     }
781   }
782   return rstr
783 }
784 func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
785   ustrv := []string{}
786   for _,v := range strv {
787     ustrv = append(ustrv,unescapeWhiteSP(v))
788   }
789   return ustrv
790 }
791
792 // <a name="comexpansion">str-expansion</a>
793 // - this should be a macro processor
794 func strsubst(gshCtx *GshContext,str string,histonly bool) string {
795   rbuff := []byte{}
796   if false {
797     //@@U Unicode should be cared as a character
798     return str
799   }
800   //rstr := ""
801   inEsc := 0 // escape characer mode
802   for i := 0; i < len(str); i++ {
803     //fmt.Printf("--D--Subst %v:%v\n",i,str[i:])
804     ch := str[i]
805     if inEsc == 0 {
806       if ch == '|' {
807         //leng,xrstr := substHistory(gshCtx,str,i,rstr)
808         leng,rs := substHistory(gshCtx,str,i,"")
809         if 0 < leng {
810           //_,rs := substHistory(gshCtx,str,i,"")
811           rbuff = append(rbuff,[]byte(rs)...)
812           i += leng
813           //rstr = xrstr
814           continue
815         }
816       }
817       switch ch {
818         case '\\': inEsc = '\\'; continue
819         case '%': inEsc = '%'; continue
820         case '$':
821         }
822     }
823     switch inEsc {
824     case '\\':
825       switch ch {
826         case '\\': ch = '\\'
827         case 's': ch = ' '
828         case 't': ch = '\t'
829         case 'r': ch = '\r'
830         case 'n': ch = '\n'
831         case 'z': inEsc = 0; continue // empty, to be ignored
832       }
833     case '%':
834       switch {
835         case ch == '%': ch = '%'
836         case ch == 'T':
837           //rstr = rstr + time.Now().Format(time.Stamp)
838         }
839     case '$':
840       rs := time.Now().Format(time.Stamp)
841       rbuff = append(rbuff,[]byte(rs)...)
842       inEsc = 0
843       continue;
844     default:
845       // postpone the interpretation
846       //rstr = rstr + "%" + string(ch)
847       rbuff = append(rbuff,ch)
848       inEsc = 0
849       continue;
850     }
851     inEsc = 0
852   }
853   //rstr = rstr + string(ch)
854   rbuff = append(rbuff,ch)
855 }
856 //fmt.Printf("--D--subst(%s)(%s)\n",str,string(rbuff))
857 return string(rbuff)
858 //return rstr
859 }
860
861 func showFileInfo(path string, opts []string) {
862   if isin("-l",opts) || isin("-ls",opts) {
863     fi, err := os.Stat(path)
864     if err != nil {
865       fmt.Printf("----- ((%v))",err)
866     }else{
867       mod := fi.ModTime()
868       date := mod.Format(time.Stamp)
869       fmt.Printf("%v %8v %s ",fi.Mode(),fi.Size(),date)
870     }
871   }
872 }

```

```

869     }
870     fmt.Printf("%s",path)
871     if !isin("-sp",opts) {
872         fmt.Printf(" ")
873     }else
874     if ! isin("-n",opts) {
875         fmt.Printf("\n")
876     }
877 }
878 func userHomeDir()(string,bool){
879     /*
880     homedir,_ = os.UserHomeDir() // not implemented in older Golang
881     */
882     homedir,found := os.LookupEnv("HOME")
883     //fmt.Printf("--I-- HOME=%v(%v)\n",homedir,found)
884     if !found {
885         return "/tmp",found
886     }
887     return homedir,found
888 }
889
890 func toFullpath(path string) (fullpath string) {
891     if path[0] == '/' {
892         return path
893     }
894     pathv := strings.Split(path,DIRSEP)
895     switch {
896     case pathv[0] == ".":
897         pathv[0],_ = os.Getwd()
898     case pathv[0] == "..": // all ones should be interpreted
899         cwd,_ := os.Getwd()
900         ppathv := strings.Split(cwd,DIRSEP)
901         pathv[0] = strings.Join(ppathv,DIRSEP)
902     case pathv[0] == "~":
903         pathv[0],_ = userHomeDir()
904     default:
905         cwd,_ := os.Getwd()
906         pathv[0] = cwd + DIRSEP + pathv[0]
907     }
908     return strings.Join(pathv,DIRSEP)
909 }
910
911 func IsRegFile(path string)(bool){
912     fi, err := os.Stat(path)
913     if err == nil {
914         fm := fi.Mode()
915         return fm.IsRegular();
916     }
917     return false
918 }
919
920 // <a name="encode">Encode / Decode</a>
921 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
922 func (gshCtx *GshContext)Enc(argv[]string){
923     file := os.Stdin
924     buff := make([]byte,LINESIZE)
925     li := 0
926     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
927     for li = 0; ; li++ {
928         count, err := file.Read(buff)
929         if count <= 0 {
930             break
931         }
932         if err != nil {
933             break
934         }
935         encoder.Write(buff[0:count])
936     }
937     encoder.Close()
938 }
939 func (gshCtx *GshContext)Dec(argv[]string){
940     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
941     li := 0
942     buff := make([]byte,LINESIZE)
943     for li = 0; ; li++ {
944         count, err := decoder.Read(buff)
945         if count <= 0 {
946             break
947         }
948         if err != nil {
949             break
950         }
951         os.Stdout.Write(buff[0:count])
952     }
953 }
954 // lnspl [N] [-crlf][-C \\\]
955 func (gshCtx *GshContext)SplitLine(argv[]string){
956     strRep := isin("-str",argv) // "...+
957     reader := bufio.NewReaderSize(os.Stdin,64*1024)
958     ni := 0
959     toi := 0
960     for ni = 0; ; ni++ {
961         line, err := reader.ReadString('\n')
962         if len(line) <= 0 {
963             if err != nil {
964                 fmt.Fprintf(os.Stderr,"--I-- lnspl %d to %d (%v)\n",ni,toi,err)
965                 break
966             }
967         }
968         off := 0
969         ilen := len(line)
970         remlen := len(line)
971         if strRep { os.Stdout.Write([]byte("\n")) }
972         for oi := 0; 0 < remlen; oi++ {
973             olen := remlen
974             addnl := false
975             if 72 < olen {
976                 olen = 72
977                 addnl = true
978             }
979             fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d/%d\n",
980                 toi,ni,oi,off,olen,remlen,ilen)
981             toi += 1
982             os.Stdout.Write([]byte(line[0:olen]))
983             if addnl {
984                 if strRep {
985                     os.Stdout.Write([]byte("\n\n"))
986                 }else{
987                     //os.Stdout.Write([]byte("\r\n"))
988                     os.Stdout.Write([]byte("\n"))
989                     os.Stdout.Write([]byte("\n"))
990                 }
991             }
992             line = line[olen:]

```



```

993         off += olen
994         remlen -= olen
995     }
996     if strRep { os.Stdout.Write([]byte("\n")) }
997 }
998 fmt.Fprintf(os.Stderr, "--I-- lnsq %d to %d\n", ni, toi)
999 }
1000
1001 // CRC32 <a href="http://golang.jp/pkg/hash-crc32">crc32</a>
1002 // 1 0000 0100 1100 0001 0001 1101 1011 0111
1003 var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
1004 var CRC32IEEE uint32 = uint32(0xEDB88320)
1005 func byteCRC32add(crc uint32, str[]byte, len uint64)(uint32){
1006     var oi uint64
1007     for oi = 0; oi < len; oi++ {
1008         var oct = str[oi]
1009         for bi := 0; bi < 8; bi++ {
1010             //fprintf(stderr, "--CRC32 %d %X (%d.%d)\n", crc, oct, oi, bi)
1011             ovf1 := (crc & 0x80000000) != 0
1012             ovf2 := (oct & 0x80) != 0
1013             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
1014             oct <<= 1
1015             crc <<= 1
1016             if ovf { crc ^= CRC32UNIX }
1017         }
1018     }
1019     //fprintf(stderr, "--CRC32 return %d %d\n", crc, len)
1020     return crc;
1021 }
1022 func byteCRC32end(crc uint32, len uint64)(uint32){
1023     var slen = make([]byte, 4)
1024     var li = 0
1025     for li = 0; li < 4; {
1026         slen[li] = byte(len)
1027         li += 1
1028         len >>= 8
1029         if( len == 0 ){
1030             break
1031         }
1032     }
1033     crc = byteCRC32add(crc, slen, uint64(li))
1034     crc ^= 0xFFFFFFFF
1035     return crc
1036 }
1037 func strCRC32(str string, len uint64)(crc uint32){
1038     crc = byteCRC32add(0, []byte(str), len)
1039     crc = byteCRC32end(crc, len)
1040     //fprintf(stderr, "--CRC32 %d %d\n", crc, len)
1041     return crc
1042 }
1043 func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
1044     var slen = make([]byte, 4)
1045     var li = 0
1046     for li = 0; li < 4; {
1047         slen[li] = byte(len & 0xFF)
1048         li += 1
1049         len >>= 8
1050         if( len == 0 ){
1051             break
1052         }
1053     }
1054     crc = crc32.Update(crc, table, slen)
1055     crc ^= 0xFFFFFFFF
1056     return crc
1057 }
1058
1059 func (gsh*GshContext)xCksum(path string, argv[]string, sum*Checksum)(int64){
1060     if isin("-type/f", argv) && !IsRegFile(path){
1061         return 0
1062     }
1063     if isin("-type/d", argv) && IsRegFile(path){
1064         return 0
1065     }
1066     file, err := os.OpenFile(path, os.O_RDONLY, 0)
1067     if err != nil {
1068         fmt.Printf("--E-- cksum %v (%v)\n", path, err)
1069         return -1
1070     }
1071     defer file.Close()
1072     if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n", path, argv) }
1073
1074     bi := 0
1075     var buff = make([]byte, 32*1024)
1076     var total int64 = 0
1077     var initTime = time.Time{}
1078     if sum.Start == initTime {
1079         sum.Start = time.Now()
1080     }
1081     for bi = 0; ; bi++ {
1082         count, err := file.Read(buff)
1083         if count <= 0 || err != nil {
1084             break
1085         }
1086         if (sum.SumType & SUM_SUM64) != 0 {
1087             s := sum.Sum64
1088             for _, c := range buff[0:count] {
1089                 s += uint64(c)
1090             }
1091             sum.Sum64 = s
1092         }
1093         if (sum.SumType & SUM_UNIXFILE) != 0 {
1094             sum.Crc32Val = byteCRC32add(sum.Crc32Val, buff, uint64(count))
1095         }
1096         if (sum.SumType & SUM_CRCIEEE) != 0 {
1097             sum.Crc32Val = crc32.Update(sum.Crc32Val, &sum.Crc32Table, buff[0:count])
1098         }
1099         // <a href="https://en.wikipedia.org/wiki/BSD_checksum">BSD checksum</a>
1100         if (sum.SumType & SUM_SUM16_BSD) != 0 {
1101             s := sum.Sum16
1102             for _, c := range buff[0:count] {
1103                 s = (s >> 1) + ((s & 1) << 15)
1104                 s += int(c)
1105                 s &= 0xFFFF
1106                 //fmt.Printf("BSDsum: %d[%d] %d\n", sum.Size+int64(i), i, s)
1107             }
1108             sum.Sum16 = s
1109         }
1110         if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1111             for bj := 0; bj < count; bj++ {
1112                 sum.Sum16 += int(buff[bj])
1113             }
1114         }
1115         total += int64(count)
1116     }

```

```

1117     sum.Done = time.Now()
1118     sum.Files += 1
1119     sum.Size += total
1120     if !isin("-",argv) {
1121         fmt.Printf("%v ",total)
1122     }
1123     return 0
1124 }
1125
1126 // <a name="grep">grep</a>
1127 // "lines", "lin" or "lnp" for "(text) line processor" or "scanner"
1128 // a*,!ab,c, ... sequential combination of patterns
1129 // what "LINE" is should be definable
1130 // generic line-by-line processing
1131 // grep [-v]
1132 // cat -n -v
1133 // uniq [-c]
1134 // tail -f
1135 // sed s/x/y/ or awk
1136 // grep with line count like wc
1137 // rewrite contents if specified
1138 func (gsh*GshContext)Xgrep(path string,rxpv[]string)(int){
1139     file, err := os.OpenFile(path,os.O_RDONLY,0)
1140     if err != nil {
1141         fmt.Printf("--E-- grep %v (%v)\n",path,err)
1142         return -1
1143     }
1144     defer file.Close()
1145     if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n",path,rxpv) }
1146     //reader := bufio.NewReaderSize(file,LINESIZE)
1147     reader := bufio.NewReaderSize(file,80)
1148     li := 0
1149     found := 0
1150     for li = 0; ; li++ {
1151         line, err := reader.ReadString('\n')
1152         if len(line) <= 0 {
1153             break
1154         }
1155         if 150 < len(line) {
1156             // maybe binary
1157             break;
1158         }
1159         if err != nil {
1160             break
1161         }
1162         if 0 <= strings.Index(string(line),rxpv[0]) {
1163             found += 1
1164             fmt.Printf("%s:%d: %s",path,li,line)
1165         }
1166     }
1167     //fmt.Printf("total %d lines %s\n",li,path)
1168     //if( 0 < found ){ fmt.Printf("((found %d lines %s))\n",found,path); }
1169     return found
1170 }
1171
1172 // <a name="finder">Finder</a>
1173 // finding files with it name and contents
1174 // file names are ORed
1175 // show the content with %x fmt list
1176 // ls -R
1177 // tar command by adding output
1178 type fileSum struct {
1179     Err int64 // access error or so
1180     Size int64 // content size
1181     DupSize int64 // content size from hard links
1182     Blocks int64 // number of blocks (of 512 bytes)
1183     DupBlocks int64 // Blocks pointed from hard links
1184     HLinks int64 // hard links
1185     Words int64
1186     Lines int64
1187     Files int64
1188     Dirs int64 // the num. of directories
1189     SymLink int64
1190     Flats int64 // the num. of flat files
1191     MaxDepth int64
1192     MaxNamlen int64 // max. name length
1193     nextRepo time.Time
1194 }
1195 func showFusage(dir string,fusage *fileSum){
1196     bsume := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1197     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1198
1199     fmt.Printf("%v: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
1200         dir,
1201         fusage.Files,
1202         fusage.Dirs,
1203         fusage.SymLink,
1204         fusage.HLinks,
1205         float64(fusage.Size)/1000000.0,bsume);
1206 }
1207 const (
1208     S_IFMT = 0170000
1209     S_IFCHR = 0020000
1210     S_IFDIR = 0040000
1211     S_IFREG = 0100000
1212     S_IFLNK = 0120000
1213     S_IFSOCK = 0140000
1214 )
1215 func cumPinfo(fsum *fileSum, path string, staterr error, fstat syscall.Stat_t, argv[]string,verb bool)(*fileSum){
1216     now := time.Now()
1217     if time.Second <= now.Sub(fsum.nextRepo) {
1218         if !fsum.nextRepo.IsZero(){
1219             tstamp := now.Format(time.Stamp)
1220             showFusage(tstamp,fsum)
1221         }
1222         fsum.nextRepo = now.Add(time.Second)
1223     }
1224     if staterr != nil {
1225         fsum.Err += 1
1226         return fsum
1227     }
1228     fsum.Files += 1
1229     if 1 < fstat.Nlink {
1230         // must count only once...
1231         // at least ignore ones in the same directory
1232         //if finfo.Mode().IsRegular() {
1233         if (fstat.Mode & S_IFMT) == S_IFREG {
1234             fsum.HLinks += 1
1235             fsum.DupBlocks += int64(fstat.Blocks)
1236             //fmt.Printf("---Dup HardLink %v %s\n",fstat.Nlink,path)
1237         }
1238     }
1239     //fsum.Size += finfo.Size()
1240     fsum.Size += fstat.Size

```

```

1241 fsum.Blocks += int64(fstat.Blocks)
1242 //if verb { fmt.Printf("%8dBlk) %s", fstat.Blocks/2, path) }
1243 if isin("-ls", argv){
1244     //if verb { fmt.Printf("%4d %8d ", fstat.Blksize, fstat.Blocks) }
1245     // fmt.Printf("%d\t", fstat.Blocks/2)
1246 }
1247 //if finfo.IsDir()
1248 if (fstat.Mode & S_IFMT) == S_IFDIR {
1249     fsum.Dirs += 1
1250 }
1251 //if (finfo.Mode() & os.ModeSymlink) != 0
1252 if (fstat.Mode & S_IFMT) == S_IFLNK {
1253     //if verb { fmt.Printf("symlink(%v,%s)\n", fstat.Mode, finfo.Name()) }
1254     //{ fmt.Printf("symlink(%0,%s)\n", fstat.Mode, finfo.Name()) }
1255     fsum.Symlink += 1
1256 }
1257 }
1258 }
1259 func (gsh*GshContext)xxFindEntv(depth int, total *fileSum, dir string, dstat syscall.Stat_t, ei int, entv []string, npatv[]string, argv[]string)(*fileSum){
1260     nols := isin("-grep", argv)
1261     // sort entv
1262     /*
1263     if isin("-t", argv){
1264         sort.Slice(filev, func(i, j int) bool {
1265             return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
1266         })
1267     }
1268     */
1269     /*
1270     if isin("-u", argv){
1271         sort.Slice(filev, func(i, j int) bool {
1272             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1273         })
1274     }
1275     if isin("-U", argv){
1276         sort.Slice(filev, func(i, j int) bool {
1277             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1278         })
1279     }
1280     */
1281     /*
1282     if isin("-s", argv){
1283         sort.Slice(filev, func(i, j int) bool {
1284             return filev[j].Size() < filev[i].Size()
1285         })
1286     }
1287     */
1288     for _, filename := range entv {
1289         for _, npat := range npatv {
1290             match := true
1291             if npat == "*" {
1292                 match = true
1293             } else {
1294                 match, _ = filepath.Match(npat, filename)
1295             }
1296             path := dir + DIRSEP + filename
1297             if !match {
1298                 continue
1299             }
1300             var fstat syscall.Stat_t
1301             staterr := syscall.Lstat(path, &fstat)
1302             if staterr != nil {
1303                 if !isin("-w", argv){fmt.Printf("ufind: %v\n", staterr) }
1304                 continue;
1305             }
1306             if isin("-du", argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
1307                 // should not show size of directory in "-du" mode ...
1308             } else
1309             if !nols && !isin("-s", argv) && (!isin("-du", argv) || isin("-a", argv)) {
1310                 if isin("-du", argv) {
1311                     fmt.Printf("%d\t", fstat.Blocks/2)
1312                 }
1313                 showFileInfo(path, argv)
1314             }
1315             if true { // && isin("-du", argv)
1316                 total = cumFinfo(total, path, staterr, fstat, argv, false)
1317             }
1318             /*
1319             if isin("-wc", argv) {
1320                 }
1321             */
1322             if gsh.lastCheckSum.SumType != 0 {
1323                 gsh.xCksum(path, argv, &gsh.lastCheckSum);
1324             }
1325             x := isinX("-grep", argv); // -grep will be convenient like -ls
1326             if 0 < x && x+1 <= len(argv) { // -grep will be convenient like -ls
1327                 if IsRegFile(path){
1328                     found := gsh.xGrep(path, argv[x+1:])
1329                     if 0 < found {
1330                         foundv := gsh.CmdCurrent.FoundFile
1331                         if len(foundv) < 10 {
1332                             gsh.CmdCurrent.FoundFile =
1333                                 append(gsh.CmdCurrent.FoundFile, path)
1334                         }
1335                     }
1336                 }
1337             }
1338             if !isin("-r0", argv) { // -d 0 in du, -depth n in find
1339                 //total.Depth += 1
1340                 if (fstat.Mode & S_IFMT) == S_IFLNK {
1341                     continue
1342                 }
1343                 if dstat.Rdev != fstat.Rdev {
1344                     fmt.Printf("--I-- don't follow differnet device %v(%v) %v(%v)\n",
1345                         dir, dstat.Rdev, path, fstat.Rdev)
1346                 }
1347                 if (fstat.Mode & S_IFMT) == S_IFDIR {
1348                     total = gsh.xxFind(depth+1, total, path, npatv, argv)
1349                 }
1350             }
1351         }
1352     }
1353     return total
1354 }
1355 func (gsh*GshContext)xxFind(depth int, total *fileSum, dir string, npatv[]string, argv[]string)(*fileSum){
1356     nols := isin("-grep", argv)
1357     dirfile, oerr := os.OpenFile(dir, os.O_RDONLY, 0)
1358     if oerr == nil {
1359         //fmt.Printf("--I-- %v(%v)[%d]\n", dir, dirfile, dirfile.Fd())
1360         defer dirfile.Close()
1361     } else {
1362     }
1363 }
1364 prev := *total

```

```

1365 var dstat syscall.Stat_t
1366 staterr := syscall.Lstat(dir,dstat) // should be lstat
1367
1368 if staterr != nil {
1369     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",staterr) }
1370     return total
1371 }
1372 //filev,err := ioutil.ReadDir(dir)
1373 //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1374 /*
1375 if err != nil {
1376     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1377     return total
1378 }
1379 */
1380 if depth == 0 {
1381     total = cumFinfo(total,dir,staterr,dstat,argv,true)
1382     if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1383         showFileInfo(dir,argv)
1384     }
1385 }
1386 // it it is not a directory, just scan it and finish
1387
1388 for ei := 0; ; ei++ {
1389     entv,rderr := dirfile.Readdirnames(8*1024)
1390     if len(entv) == 0 || rderr != nil {
1391         //if rderr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),rderr) }
1392         break
1393     }
1394     if 0 < ei {
1395         fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
1396     }
1397     total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npats,argv)
1398 }
1399 if isin("-du",argv) {
1400     // if in "du" mode
1401     fmt.Printf("%d\t%s\n",total.Blocks-prev.Blocks)/2,dir)
1402 }
1403 }
1404 }
1405
1406 // {ufind|fu|ls} [Files] [-- Names] [-- Expressions]
1407 // Files is "." by default
1408 // Names is "*" by default
1409 // Expressions is "-print" by default for "ufind", or -du for "fu" command
1410 func (gsh*GshContext)xFind(argv[]string){
1411     if 0 < len(argv) && strBegins(argv[0],"?"){
1412         showFound(gsh,argv)
1413         return
1414     }
1415     if isin("-cksum",argv) || isin("-sum",argv) {
1416         gsh.lastCheckSum = CheckSum{}
1417         if isin("-sum",argv) && isin("-add",argv) {
1418             gsh.lastCheckSum.SumType |= SUM_SUM64
1419         }else
1420         if isin("-sum",argv) && isin("-size",argv) {
1421             gsh.lastCheckSum.SumType |= SUM_SIZE
1422         }else
1423         if isin("-sum",argv) && isin("-bsd",argv) {
1424             gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
1425         }else
1426         if isin("-sum",argv) && isin("-sysv",argv) {
1427             gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
1428         }else
1429         if isin("-sum",argv) {
1430             gsh.lastCheckSum.SumType |= SUM_SUM64
1431         }
1432         if isin("-unix",argv) {
1433             gsh.lastCheckSum.SumType |= SUM_UNIXFILE
1434             gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32UNIX)
1435         }
1436         if isin("-ieee",argv){
1437             gsh.lastCheckSum.SumType |= SUM_CRCIEEE
1438             gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32IEEE)
1439         }
1440         gsh.lastCheckSum.RusgAtStart = Getrusagev()
1441     }
1442     var total = fileSum{}
1443     npats := []string{}
1444     for _,v := range argv {
1445         if 0 < len(v) && v[0] != '-' {
1446             npats = append(npats,v)
1447         }
1448         if v == "/" { break }
1449         if v == "--" { break }
1450         if v == "-grep" { break }
1451         if v == "-ls" { break }
1452     }
1453     if len(npats) == 0 {
1454         npats = []string{"*"}
1455     }
1456     cwd := "."
1457     // if to be fullpath :: cwd, _ := os.Getwd()
1458     if len(npats) == 0 { npats = []string{"*"} }
1459     fusage := gsh.xxFind(0,&total,cwd,npats,argv)
1460     if gsh.lastCheckSum.SumType != 0 {
1461         var sumi uint64 = 0
1462         sum := &gsh.lastCheckSum
1463         if (sum.SumType & SUM_SIZE) != 0 {
1464             sumi = uint64(sum.Size)
1465         }
1466         if (sum.SumType & SUM_SUM64) != 0 {
1467             sumi = sum.Sum64
1468         }
1469         if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1470             s := uint32(sum.Sum16)
1471             r := (s & 0xFFFF) + ((s & 0xFFFFFFFF) >> 16)
1472             s = (r & 0xFFFF) + (r >> 16)
1473             sum.Crc32Val = uint32(s)
1474             sumi = uint64(s)
1475         }
1476         if (sum.SumType & SUM_SUM16_BSD) != 0 {
1477             sum.Crc32Val = uint32(sum.Sum16)
1478             sumi = uint64(sum.Sum16)
1479         }
1480         if (sum.SumType & SUM_UNIXFILE) != 0 {
1481             sum.Crc32Val = byteCRC32end(sum.Crc32Val,uint64(sum.Size))
1482             sumi = uint64(byteCRC32end(sum.Crc32Val,uint64(sum.Size)))
1483         }
1484     }
1485     if 1 < sum.Files {
1486         fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
1487             sumi,sum.Size,
1488             abssize(sum.Size),sum.Files,
1489             abssize(sum.Size/sum.Files))

```

```

1489         }else{
1490             fmt.Printf("%v %v %v\n",
1491                 sumi,sum.Size,npats[0])
1492         }
1493     }
1494     if !isin("-grep",argv) {
1495         showFusage("total",fusage)
1496     }
1497     if !isin("-s",argv){
1498         hits := len(gsh.CmdCurrent.FoundFile)
1499         if 0 < hits {
1500             fmt.Printf("--I-- %d files hits // can be refered with !&df\n",
1501                 hits,len(gsh.CommandHistory))
1502         }
1503     }
1504     if gsh.lastCheckSum.SumType != 0 {
1505         if isin("-ru",argv) {
1506             sum := gsh.lastCheckSum
1507             sum.Done = time.Now()
1508             gsh.lastCheckSum.RusgAtEnd = Getrusagev()
1509             elps := sum.Done.Sub(sum.Start)
1510             fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\r\n",
1511                 sum.Size,abssize(sum.Size),sum.Files,abssize(sum.Size/sum.Files))
1512             nanos := int64(elps)
1513             fmt.Printf("--cksum-time: %v/total, %v/file, %i.if files/s, %v\r\n",
1514                 abftime(nanos),
1515                 abftime(nanos/sum.Files),
1516                 (float64(sum.Files)*1000000000.0)/float64(nanos),
1517                 abbspeed(sum.Size,nanos))
1518             diff := RusageSubv(sum.RusgAtEnd,sum.RusgAtStart)
1519             fmt.Printf("--cksum-rusg: %v\n",sRusagef("",argv,diff))
1520         }
1521     }
1522     return
1523 }
1524
1525 func showFiles(files[]string){
1526     sp := ""
1527     for i,file := range files {
1528         if 0 < i { sp = " " } else { sp = "" }
1529         fmt.Printf(sp+"%s",escapeWhitesP(file))
1530     }
1531 }
1532 func showFound(gshCtx *GshContext, argv[]string){
1533     for i,v := range gshCtx.CommandHistory {
1534         if 0 < len(v.FoundFile) {
1535             fmt.Printf("!\%d (%d) ",i,len(v.FoundFile))
1536             if isin("-ls",argv){
1537                 fmt.Printf("\n")
1538                 for _,file := range v.FoundFile {
1539                     fmt.Printf(" ") //sub number?
1540                     showFileInfo(file,argv)
1541                 }
1542             }else{
1543                 showFiles(v.FoundFile)
1544                 fmt.Printf("\n")
1545             }
1546         }
1547     }
1548 }
1549
1550 func showMatchFile(filev []os.FileInfo, npat,dir string, argv[]string)(string,bool){
1551     fname := ""
1552     found := false
1553     for _,v := range filev {
1554         match, _ := filepath.Match(npat,(v.Name()))
1555         if match {
1556             fname = v.Name()
1557             found = true
1558             //fmt.Printf("!\%d] %s\n",i,v.Name())
1559             showIfExecutable(fname,dir,argv)
1560         }
1561     }
1562     return fname,found
1563 }
1564 func showIfExecutable(name,dir string,argv[]string)(ffullpath string,ffound bool){
1565     var fullpath string
1566     if strBegins(name,DIRSEP){
1567         fullpath = name
1568     }else{
1569         fullpath = dir + DIRSEP + name
1570     }
1571     fi, err := os.Stat(fullpath)
1572     if err != nil {
1573         fullpath = dir + DIRSEP + name + ".go"
1574         fi, err = os.Stat(fullpath)
1575     }
1576     if err == nil {
1577         fm := fi.Mode()
1578         if fm.IsRegular() {
1579             // R_OK=4, W_OK=2, X_OK=1, F_OK=0
1580             if syscall.Access(fullpath,5) == nil {
1581                 ffullpath = fullpath
1582                 ffound = true
1583                 if ! isin("-s", argv) {
1584                     showFileInfo(fullpath,argv)
1585                 }
1586             }
1587         }
1588     }
1589     return ffullpath, ffound
1590 }
1591 func which(list string, argv []string) (fullpathv []string, itis bool){
1592     if len(argv) <= 1 {
1593         fmt.Printf("Usage: which comand [-s] [-a] [-ls]\n")
1594         return []string(""), false
1595     }
1596     path := argv[1]
1597     if strBegins(path,"/") {
1598         // should check if excoecutable?
1599         _,exOK := showIfExecutable(path,"",argv)
1600         fmt.Printf("--D-- %v exOK=%v\n",path,exOK)
1601         return []string(path),exOK
1602     }
1603     pathenv, efound := os.LookupEnv(list)
1604     if ! efound {
1605         fmt.Printf("--E-- which: no \"%s\" environment\n",list)
1606         return []string(""), false
1607     }
1608     showall := isin("-a",argv) || 0 <= strings.Index(path,"*")
1609     dirv := strings.Split(pathenv,PATHSEP)
1610     ifound := false
1611     ffullpath := path
1612     for _, dir := range dirv {

```

```

1613     if 0 <= strings.Index(path,"*") { // by wild-card
1614         list,_ := ioutil.ReadDir(dir)
1615         ffullpath, ffound = showMatchFile(list,path,dir,argv)
1616     }else{
1617         ffullpath, ffound = showIfExecutable(path,dir,argv)
1618     }
1619     //if ffound && !isin("-a", argv) {
1620     if ffound && !showall {
1621         break;
1622     }
1623 }
1624 return []string{ffullpath}, ffound
1625 }
1626
1627 func stripLeadingWSParg(argv []string) ([]string){
1628     for ; 0 < len(argv); {
1629         if len(argv[0]) == 0 {
1630             argv = argv[1:]
1631         }else{
1632             break
1633         }
1634     }
1635     return argv
1636 }
1637 func xEval(argv []string, nlend bool){
1638     argv = stripLeadingWSParg(argv)
1639     if len(argv) == 0 {
1640         fmt.Printf("eval [%%format] [Go-expression]\n")
1641         return
1642     }
1643     pfmt := "%v"
1644     if argv[0][0] == '%' {
1645         pfmt = argv[0]
1646         argv = argv[1:]
1647     }
1648     if len(argv) == 0 {
1649         return
1650     }
1651     gocode := strings.Join(argv, " ");
1652     //fmt.Printf("eval [%v] [%v]\n",pfmt,gocode)
1653     fset := token.NewFileSet()
1654     rval, _ := types.Eval(fset,nil,token.NoPos,gocode)
1655     fmt.Printf(pfmt,rval.Value)
1656     if nlend { fmt.Printf("\n") }
1657 }
1658
1659 func getval(name string) (found bool, val int) {
1660     /* should expand the name here */
1661     if name == "gsh.pid" {
1662         return true, os.Getpid()
1663     }else
1664     if name == "gsh.ppid" {
1665         return true, os.Getppid()
1666     }
1667     return false, 0
1668 }
1669
1670 func echo(argv []string, nlend bool){
1671     for ai := 1; ai < len(argv); ai++ {
1672         if 1 < ai {
1673             fmt.Printf(" ");
1674         }
1675         arg := argv[ai]
1676         found, val := getval(arg)
1677         if found {
1678             fmt.Printf("%d",val)
1679         }else{
1680             fmt.Printf("%s",arg)
1681         }
1682     }
1683     if nlend {
1684         fmt.Printf("\n");
1685     }
1686 }
1687
1688 func resfile() string {
1689     return "gsh.tmp"
1690 }
1691 //var resF *File
1692 func resmap() {
1693     //_, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
1694     // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
1695     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
1696     if err != nil {
1697         fmt.Printf("refF could not open: %s\n",err)
1698     }else{
1699         fmt.Printf("refF opened\n")
1700     }
1701 }
1702
1703 // @2020-0821
1704 func gshScanArg(str string,strip int)(argv []string){
1705     var si = 0
1706     var sb = 0
1707     var inBracket = 0
1708     var arg1 = make([]byte,LINESIZE)
1709     var ax = 0
1710     debug := false
1711
1712     for ; si < len(str); si++ {
1713         if str[si] != ' ' {
1714             break
1715         }
1716     }
1717     sb = si
1718     for ; si < len(str); si++ {
1719         if sb <= si {
1720             if debug {
1721                 fmt.Printf("--Da- +%d %2d-%2d %s ... %s\n",
1722                     inBracket,sb,si,arg1[0:ax],str[si:])
1723             }
1724         }
1725         ch := str[si]
1726         if ch == '{' {
1727             inBracket += 1
1728             if 0 < strip && inBracket <= strip {
1729                 //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
1730                 continue
1731             }
1732         }
1733         if 0 < inBracket {
1734             if ch == '}' {
1735                 inBracket -= 1
1736                 if 0 < strip && inBracket < strip {

```

```

1737         //fmt.Printf("stripLEV %d < %d?\n",inBracket,strip)
1738         continue
1739     }
1740     }
1741     argl[ax] = ch
1742     ax += 1
1743     continue
1744 }
1745 if str[si] == ' ' {
1746     argv = append(argv,string(argl[0:ax]))
1747     if debug {
1748         fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
1749             -1+len(argv),sb,si,str[sb:si],string(str[si:]))
1750     }
1751     sb = si+1
1752     ax = 0
1753     continue
1754 }
1755 argl[ax] = ch
1756 ax += 1
1757 }
1758 if sb < si {
1759     argv = append(argv,string(argl[0:ax]))
1760     if debug {
1761         fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
1762             -1+len(argv),sb,si,string(argl[0:ax]),string(str[si:]))
1763     }
1764 }
1765 if debug {
1766     fmt.Printf("--Da- %d [%s] => [%d]%v\n",strip,str,len(argv),argv)
1767 }
1768 return argv
1769 }
1770
1771 // should get stderr (into tmpfile ?) and return
1772 func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
1773     var pv = []int{-1,-1}
1774     syscall.Pipe(pv)
1775
1776     xarg := gshScanArg(name,1)
1777     name = strings.Join(xarg," ")
1778
1779     pin = os.NewFile(uintptr(pv[0]),"StdoutOf-"+name+"")
1780     pout = os.NewFile(uintptr(pv[1]),"StdinOf-"+name+"")
1781     fdix := 0
1782     dir := "?"
1783     if mode == "r" {
1784         dir = "<"
1785         fdix = 1 // read from the stdout of the process
1786     }else{
1787         dir = ">"
1788         fdix = 0 // write to the stdin of the process
1789     }
1790     gshPA := gsh.gshPA
1791     savfd := gshPA.Files[fdix]
1792
1793     var fd uintptr = 0
1794     if mode == "r" {
1795         fd = pout.Fd()
1796         gshPA.Files[fdix] = pout.Fd()
1797     }else{
1798         fd = pin.Fd()
1799         gshPA.Files[fdix] = pin.Fd()
1800     }
1801
1802     // should do this by Goroutine?
1803     if false {
1804         fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
1805         fmt.Printf("--RED1 [%d,%d,%d]->[%d,%d,%d]\n",
1806             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
1807             pin.Fd(),pout.Fd(),pout.Fd())
1808     }
1809     savi := os.Stdin
1810     savo := os.Stdout
1811     save := os.Stderr
1812     os.Stdin = pin
1813     os.Stdout = pout
1814     os.Stderr = pout
1815     gsh.Background = true
1816     gsh.gshelllh(name)
1817     gsh.Background = false
1818     os.Stdin = savi
1819     os.Stdout = savo
1820     os.Stderr = save
1821
1822     gshPA.Files[fdix] = savfd
1823     return pin,pout,false
1824 }
1825
1826 // <a name="ex-commands">External commands</a>
1827 func (gsh*GshContext)excommand(exec bool,argv []string)(notf bool,exit bool) {
1828     if gsh.CmdTrace { fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }
1829
1830     gshPA := gsh.gshPA
1831     fullpathv, itis := which("PATH",[]string{"which",argv[0],"-s"})
1832     if itis == false {
1833         return true,false
1834     }
1835     fullpath := fullpathv[0]
1836     argv = unescapeWhiteSPV(argv)
1837     if 0 < strings.Index(fullpath,".go") {
1838         nargv := argv // []string{}
1839         gofullpathv, itis := which("PATH",[]string{"which","go","-s"})
1840         if itis == false {
1841             fmt.Printf("--F-- Go not found\n")
1842             return false,true
1843         }
1844         gofullpath := gofullpathv[0]
1845         nargv = []string{ gofullpath, "run", fullpath }
1846         fmt.Printf("--I-- %s (%s %s %s)\n",gofullpath,
1847             nargv[0],nargv[1],nargv[2])
1848         if exec {
1849             syscall.Exec(gofullpath,nargv,os.Environ())
1850         }else{
1851             pid, _ := syscall.ForkExec(gofullpath,nargv,&gshPA)
1852             if gsh.Background {
1853                 fmt.Fprintf(stderr,"--Ip- in Background pid[%d]%d(%v)\n",pid,len(argv),nargv)
1854                 gsh.BackgroundJobs = append(gsh.BackgroundJobs,pid)
1855             }else{
1856                 rusage := syscall.Rusage {}
1857                 syscall.Wait4(pid,nil,0,&rusage)
1858                 gsh.LastRusage = rusage
1859                 gsh.CmdCurrent.Rusagev[1] = rusage
1860             }
1861         }
1862     }

```

```

1861 }else{
1862     if exec {
1863         syscall.Exec(fullpath,argv,os.Environ())
1864     }else{
1865         pid, _ := syscall.ForkExec(fullpath,argv,&gshPA)
1866         //fmt.Printf("[%d]\n",pid); // '%&' to be background
1867         if gsh.BackGround {
1868             fmt.Fprintf(stderr,"--Ip- in Background pid[%d]%(v)\n",pid,len(argv),argv)
1869             gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
1870         }else{
1871             rusage := syscall.Rusage {}
1872             syscall.Wait4(pid,nil,0,&rusage);
1873             gsh.LastRusage = rusage
1874             gsh.CmdCurrent.Rusagev[1] = rusage
1875         }
1876     }
1877 }
1878 return false,false
1879 }
1880
1881 // <a name="builtin">Builtin Commands</a>
1882 func (gshCtx *GshContext) sleep(argv []string) {
1883     if len(argv) < 2 {
1884         fmt.Printf("Sleep 100ms, 100us, 100ns, ... \n")
1885         return
1886     }
1887     duration := argv[1];
1888     d, err := time.ParseDuration(duration)
1889     if err != nil {
1890         d, err = time.ParseDuration(duration+"s")
1891         if err != nil {
1892             fmt.Printf("duration ? %s (%s)\n",duration,err)
1893             return
1894         }
1895     }
1896     //fmt.Printf("Sleep %v\n",duration)
1897     time.Sleep(d)
1898     if 0 < len(argv[2:]) {
1899         gshCtx.gshellv(argv[2:])
1900     }
1901 }
1902 func (gshCtx *GshContext)repeat(argv []string) {
1903     if len(argv) < 2 {
1904         return
1905     }
1906     start0 := time.Now()
1907     for ri, _ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
1908         if 0 < len(argv[2:]) {
1909             //start := time.Now()
1910             gshCtx.gshellv(argv[2:])
1911             end := time.Now()
1912             elps := end.Sub(start0);
1913             if( 1000000000 < elps ){
1914                 fmt.Printf("(repeat#%d %v)\n",ri,elps);
1915             }
1916         }
1917     }
1918 }
1919
1920 func (gshCtx *GshContext)gen(argv []string) {
1921     gshPA := gshCtx.gshPA
1922     if len(argv) < 2 {
1923         fmt.Printf("Usage: %s N\n",argv[0])
1924         return
1925     }
1926     // should br repeated by "repeat" command
1927     count, _ := strconv.Atoi(argv[1])
1928     fd := gshPA.Files[1] // Stdout
1929     file := os.NewFile(fd,"internalStdOut")
1930     fmt.Printf("--I-- Gen. Count=%d to [%d]\n",count,file.Fd())
1931     //buf := []byte{}
1932     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
1933     for gi := 0; gi < count; gi++ {
1934         file.WriteString(outdata)
1935     }
1936     //file.WriteString("\n")
1937     fmt.Printf("\n(%d B)\n",count*len(outdata));
1938     //file.Close()
1939 }
1940
1941 // <a name="rexec">Remote Execution</a> // 2020-0820
1942 func Elapsed(from time.Time)(string){
1943     elps := time.Now().Sub(from)
1944     if 1000000000 < elps {
1945         return fmt.Sprintf("[%5d.%02ds]",elps/1000000000,(elps%1000000000)/1000000)
1946     }else
1947     if 1000000 < elps {
1948         return fmt.Sprintf("[%3d.%03dms]",elps/1000000,(elps%1000000)/1000)
1949     }else{
1950         return fmt.Sprintf("[%3d.%03dus]",elps/1000,(elps%1000))
1951     }
1952 }
1953 func abftime(nanos int64)(string){
1954     if 1000000000 < nanos {
1955         return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%1000000000)/1000000)
1956     }else
1957     if 1000000 < nanos {
1958         return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
1959     }else{
1960         return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
1961     }
1962 }
1963 func abssize(size int64)(string){
1964     fsize := float64(size)
1965     if 1024*1024*1024 < size {
1966         return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
1967     }else
1968     if 1024*1024 < size {
1969         return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
1970     }else{
1971         return fmt.Sprintf("%.3fKiB",fsize/1024)
1972     }
1973 }
1974 func absze(size int64)(string){
1975     fsize := float64(size)
1976     if 1024*1024*1024 < size {
1977         return fmt.Sprintf("%.8.2fGiB",fsize/(1024*1024*1024))
1978     }else
1979     if 1024*1024 < size {
1980         return fmt.Sprintf("%.8.3fMiB",fsize/(1024*1024))
1981     }else{
1982         return fmt.Sprintf("%.8.3fKiB",fsize/1024)
1983     }
1984 }

```



```

1985 func abbspeed(totalB int64,ns int64)(string){
1986     Mbs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
1987     if 1000 <= Mbs {
1988         return fmt.Sprintf("%6.3fGB/s",Mbs/1000)
1989     }
1990     if 1 <= Mbs {
1991         return fmt.Sprintf("%6.3fMB/s",Mbs)
1992     }else{
1993         return fmt.Sprintf("%6.3fKB/s",Mbs*1000)
1994     }
1995 }
1996 func abspeed(totalB int64,ns time.Duration)(string){
1997     Mbs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
1998     if 1000 <= Mbs {
1999         return fmt.Sprintf("%6.3fGBps",Mbs/1000)
2000     }
2001     if 1 <= Mbs {
2002         return fmt.Sprintf("%6.3fMBps",Mbs)
2003     }else{
2004         return fmt.Sprintf("%6.3fKBps",Mbs*1000)
2005     }
2006 }
2007 func fileRelay(what string,in*os.File,out*os.File,size int64,bsiz int)(wcount int64){
2008     Start := time.Now()
2009     buff := make([]byte,bsiz)
2010     var total int64 = 0
2011     var rem int64 = size
2012     nio := 0
2013     Prev := time.Now()
2014     var PrevSize int64 = 0
2015
2016     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) START\n",
2017         what,absize(total),size,nio)
2018
2019     for i:= 0; ; i++ {
2020         var len = bsiz
2021         if int(rem) < len {
2022             len = int(rem)
2023         }
2024         Now := time.Now()
2025         Elps := Now.Sub(Prev);
2026         if 1000000000 < Now.Sub(Prev) {
2027             fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %s\n",
2028                 what,absize(total),size,nio,
2029                 abspeed((total-PrevSize),Elps))
2030             Prev = Now;
2031             PrevSize = total
2032         }
2033         rlen := len
2034         if in != nil {
2035             // should watch the disconnection of out
2036             rcc,err := in.Read(buff[0:rlen])
2037             if err != nil {
2038                 fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<%v\n",
2039                     what,rcc,err,in.Name())
2040                 break
2041             }
2042             rlen = rcc
2043             if string(buff[0:10]) == "(SoftEOF " {
2044                 var ecc int64 = 0
2045                 fmt.Sscanf(string(buff),"(SoftEOF %v",&ecc)
2046                 fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))/%v\n",
2047                     what,ecc,total)
2048                 if ecc == total {
2049                     break
2050                 }
2051             }
2052         }
2053
2054         wlen := rlen
2055         if out != nil {
2056             wcc,err := out.Write(buff[0:rlen])
2057             if err != nil {
2058                 fmt.Printf(Elapsed(Start)+"--En-- X: %s write(%v,%v)>%v\n",
2059                     what,wcc,err,out.Name())
2060                 break
2061             }
2062             wlen = wcc
2063         }
2064         if wlen < rlen {
2065             fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2066                 what,wlen,rlen)
2067             break;
2068         }
2069
2070         nio += 1
2071         total += int64(rlen)
2072         rem -= int64(rlen)
2073         if rem <= 0 {
2074             break
2075         }
2076     }
2077     Done := time.Now()
2078     Elps := float64(Done.Sub(Start))/1000000000 //Seconds
2079     TotalMB := float64(total)/1000000 //MB
2080     MBps := TotalMB / Elps
2081     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %v %6.3fMB/s\n",
2082         what,total,size,nio,absize(total),MBps)
2083     return total
2084 }
2085 func tcpPush(clnt *os.File){
2086     // shrink socket buffer and recover
2087     usleep(100);
2088 }
2089 func (gsh*GshContext)RexecServer(argv[]string){
2090     debug := true
2091     Start0 := time.Now()
2092     Start := Start0
2093     // if local == ":" { local = "0.0.0.0:9999" }
2094     local := "0.0.0.0:9999"
2095
2096     if 0 < len(argv) {
2097         if argv[0] == "-s" {
2098             debug = false
2099             argv = argv[1:]
2100         }
2101     }
2102     if 0 < len(argv) {
2103         argv = argv[1:]
2104     }
2105     port, err := net.ResolveTCPAddr("tcp",local);
2106     if err != nil {
2107         fmt.Printf("--En- S: Address error: %s (%s)\n",local,err)
2108         return

```

```

2109 }
2110 fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n",local);
2111 scon, err := net.ListenTCP("tcp", port)
2112 if err != nil {
2113     fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n",local,err)
2114     return
2115 }
2116
2117 reqbuf := make([]byte,LINESIZE)
2118 res := ""
2119 for {
2120     fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
2121     scon, err := scon.AcceptTCP()
2122     Start = time.Now()
2123     if err != nil {
2124         fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
2125         return
2126     }
2127     c, _ := scon.File()
2128     fd := c.Fd()
2129     ar := scon.RemoteAddr()
2130     if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
2131         local,fd,ar) }
2132     res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
2133     fmt.Fprintf(c, "%s",res)
2134     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
2135     count, err := c.Read(reqbuf)
2136     if err != nil {
2137         fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
2138             count,err,string(reqbuf))
2139     }
2140     req := string(reqbuf[:count])
2141     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
2142     reqv := strings.Split(string(req)," ")
2143     cmdv := gshScanArg(reqv[0],0)
2144     //cmdv := strings.Split(reqv[0]," ")
2145     switch cmdv[0] {
2146     case "HELO":
2147         res = fmt.Sprintf("250 %v",req)
2148     case "GET":
2149         // download {remotefile|-zN} [localfile]
2150         var dsize int64 = 32*1024*1024
2151         var bsize int = 64*1024
2152         var fname string = ""
2153         var in *os.File = nil
2154         var pseudoEOF = false
2155         if 1 < len(cmdv) {
2156             fname = cmdv[1]
2157             if strBegins(fname,"-z") {
2158                 fmt.Sscanf(fname[2:],"%d",&dsize)
2159             }else
2160             if strBegins(fname,"{") {
2161                 xin,xout,err := gsh.Popen(fname,"r")
2162                 if err {
2163                     }else{
2164                         xout.Close()
2165                         defer xin.Close()
2166                         in = xin
2167                         dsize = MaxStreamSize
2168                         pseudoEOF = true
2169                     }
2170             }else{
2171                 xin,err := os.Open(fname)
2172                 if err != nil {
2173                     fmt.Printf("--En- GET (%v)\n",err)
2174                 }else{
2175                     defer xin.Close()
2176                     in = xin
2177                     fi,_ := xin.Stat()
2178                     dsize = fi.Size()
2179                 }
2180             }
2181         }
2182         //fmt.Printf(Elapsed(Start)+"--In- GET %v:%v\n",dsize,bsize)
2183         res = fmt.Sprintf("200 %v\r\n",dsize)
2184         fmt.Fprintf(c, "%v",res)
2185         tcpPush(c); // should be separated as line in receiver
2186         fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2187         wcount := fileRelay("SendGET",in,c,dsize,bsize)
2188         if pseudoEOF {
2189             in.Close() // pipe from the command
2190             // show end of stream data (its size) by OOB?
2191             SoftEOF := fmt.Sprintf("(SoftEOF %v)",wcount)
2192             fmt.Printf(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)
2193         }
2194         tcpPush(c); // to let SoftEOF data apper at the top of received data
2195         fmt.Fprintf(c, "%v\r\n",SoftEOF)
2196         tcpPush(c); // to let SoftEOF alone in a packet (separate with 200 OK)
2197         // with client generated random
2198         //fmt.Printf("--In- L: close %v (%v)\n",in.Fd(),in.Name())
2199     }
2200     res = fmt.Sprintf("200 GET done\r\n")
2201     case "PUT":
2202         // upload {srcfile|-zN} [dstfile]
2203         var dsize int64 = 32*1024*1024
2204         var bsize int = 64*1024
2205         var fname string = ""
2206         var out *os.File = nil
2207         if 1 < len(cmdv) { // localfile
2208             fmt.Sscanf(cmdv[1],"%d",&dsize)
2209         }
2210         if 2 < len(cmdv) {
2211             fname = cmdv[2]
2212             if fname == "-" {
2213                 // nul dev
2214             }else
2215             if strBegins(fname,"{") {
2216                 xin,xout,err := gsh.Popen(fname,"w")
2217                 if err {
2218                     }else{
2219                         xin.Close()
2220                         defer xout.Close()
2221                         out = xout
2222                     }
2223             }else{
2224                 // should write to temporary file
2225                 // should suppress ^C on tty
2226                 xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2227                 //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n",fname,xout,err)
2228                 if err != nil {
2229                     fmt.Printf("--En- PUT (%v)\n",err)
2230                 }else{
2231                     out = xout
2232                 }
2233             }

```

```

2233     }
2234     fmt.Printf(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
2235         fname,local,err)
2236     }
2237     fmt.Printf(Elapsed(Start)+"--In- PUT %v (/ %v)\n",dsize,bsize)
2238     fmt.Printf(Elapsed(Start)+"--In- S: 200 %v OK\r\n",dsize)
2239     fmt.Fprintf(clnt,"200 %v OK\r\n",dsize)
2240     fileRelay("RecvPUT",clnt,out,dsize,bsize)
2241     res = fmt.Sprintf("200 PUT done\r\n")
2242     default:
2243         res = fmt.Sprintf("400 What? %v",req)
2244     }
2245     swcc,serr := clnt.Write([]byte(res))
2246     if serr != nil {
2247         fmt.Printf(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v",swcc,serr,res)
2248     }else{
2249         fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2250     }
2251     aconn.Close();
2252     clnt.Close();
2253 }
2254 }
2255 }
2256 func (gsh*GshContext)RexecClient(argv []string)(int,string){
2257     debug := true
2258     Start := time.Now()
2259     if len(argv) == 1 {
2260         return -1,"EmptyARG"
2261     }
2262     argv = argv[1:]
2263     if argv[0] == "-serv" {
2264         gsh.RexecServer(argv[1:])
2265         return 0,"Server"
2266     }
2267     remote := "0.0.0.0:9999"
2268     if argv[0][0] == '@' {
2269         remote = argv[0][1:]
2270         argv = argv[1:]
2271     }
2272     if argv[0] == "-s" {
2273         debug = false
2274         argv = argv[1:]
2275     }
2276     dport, err := net.ResolveTCPAddr("tcp",remote);
2277     if err != nil {
2278         fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
2279         return -1,"AddressError"
2280     }
2281     fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
2282     serv, err := net.DialTCP("tcp",nil,dport)
2283     if err != nil {
2284         fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n",remote,err)
2285         return -1,"CannotConnect"
2286     }
2287     if debug {
2288         al := serv.LocalAddr()
2289         fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
2290     }
2291 }
2292 req := ""
2293 res := make([]byte,LINESIZE)
2294 count,err := serv.Read(res)
2295 if err != nil {
2296     fmt.Printf("--En- S: (%3d,%v) %v",count,err,string(res))
2297 }
2298 if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res)) }
2299 }
2300 if argv[0] == "GET" {
2301     savPA := gsh.gshPA
2302     var bsize int = 64*1024
2303     req = fmt.Sprintf("%v\r\n",strings.Join(argv, " "))
2304     fmt.Printf(Elapsed(Start)+"--In- C: %v",req)
2305     fmt.Fprintf(serv,req)
2306     count,err = serv.Read(res)
2307     if err != nil {
2308     }else{
2309         var dsize int64 = 0
2310         var out *os.File = nil
2311         var out_tobeclosed *os.File = nil
2312         var fname string = ""
2313         var rcode int = 0
2314         var pid int = -1
2315         fmt.Sscanf(string(res),"%d %d",&rcode,&dsize)
2316         fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
2317         if 3 <= len(argv) {
2318             fname = argv[2]
2319             if strBegins(fname,"{") {
2320                 xin,xout,err := gsh.Popen(fname,"w")
2321                 if err {
2322                     }else{
2323                         xin.Close()
2324                         defer xout.Close()
2325                         out = xout
2326                         out_tobeclosed = xout
2327                         pid = 0 // should be its pid
2328                     }
2329                 }else{
2330                     // should write to temporary file
2331                     // should suppress ^C on tty
2332                     xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2333                     if err != nil {
2334                         fmt.Print("--En- %v\n",err)
2335                     }
2336                     out = xout
2337                     //fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
2338                 }
2339             }
2340             in,_ := serv.File()
2341             fileRelay("RecvGET",in,out,dsize,bsize)
2342             if 0 <= pid {
2343                 gsh.gshPA = savPA // recovery of Fd(), and more?
2344                 fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
2345                 out_tobeclosed.Close()
2346                 //syscall.Wait4(pid,nil,0,nil) //@@
2347             }
2348         }
2349     }else
2350     if argv[0] == "PUT" {
2351         remote,_ := serv.File()
2352         var local *os.File = nil
2353         var dsize int64 = 32*1024*1024
2354         var bsize int = 64*1024
2355         var ofile string = "-"
2356         //fmt.Printf("--I-- Rex %v\n",argv)

```

```

2357     if 1 < len(argv) {
2358         fname := argv[1]
2359         if strBegins(fname, "-z") {
2360             fmt.Sscanf(fname[2:], "%d", &dsize)
2361         } else {
2362             if strBegins(fname, "(") {
2363                 xin, xout, err := gsh.Popen(fname, "r")
2364                 if err {
2365                     } else {
2366                         xout.Close()
2367                         defer xin.Close()
2368                         //in = xin
2369                         local = xin
2370                         fmt.Printf("--In- [%d] < Upload output of %v\n",
2371                             local.Fd(), fname)
2372                         ofile = "-from."+fname
2373                         dsize = MaxStreamSize
2374                     }
2375                 } else {
2376                     xlocal, err := os.Open(fname)
2377                     if err != nil {
2378                         fmt.Printf("--En- (%s)\n", err)
2379                         local = nil
2380                     } else {
2381                         local = xlocal
2382                         fi, _ := local.Stat()
2383                         dsize = fi.Size()
2384                         defer local.Close()
2385                         //fmt.Printf("--I-- Rex in(%v / %v)\n", ofile, dsize)
2386                     }
2387                     ofile = fname
2388                     fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
2389                         fname, dsize, local, err)
2390                 }
2391             }
2392             if 2 < len(argv) && argv[2] != "" {
2393                 ofile = argv[2]
2394                 //fmt.Printf("(%d)%v B.ofile=%v\n", len(argv), argv, ofile)
2395             }
2396             //fmt.Printf(Elapsed(Start)+"--I-- Rex out(%v)\n", ofile)
2397             fmt.Printf(Elapsed(Start)+"--In- PUT %v (%v)\n", dsize, bsize)
2398             req = fmt.Sprintf("PUT %v %v\n", dsize, ofile)
2399             if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v", req) }
2400             fmt.Fprintf(serv, "%v", req)
2401             count, err = serv.Read(res)
2402             if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res[0:count])) }
2403             fileRelay("SendPUT", local, remote, dsize, bsize)
2404         } else {
2405             req = fmt.Sprintf("%v\n", strings.Join(argv, " "))
2406             if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v", req) }
2407             fmt.Fprintf(serv, "%v", req)
2408             //fmt.Printf("--In- sending RexRequest(%v)\n", len(req))
2409         }
2410         //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
2411         count, err = serv.Read(res)
2412         ress := ""
2413         if count == 0 {
2414             ress = "(nil)\n"
2415         } else {
2416             ress = string(res[:count])
2417         }
2418         if err != nil {
2419             fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v", count, err, ress)
2420         } else {
2421             fmt.Printf(Elapsed(Start)+"--In- S: %v", ress)
2422         }
2423         serv.Close()
2424         //conn.Close()
2425
2426         var stat string
2427         var rcode int
2428         fmt.Sscanf(ress, "%d %s", &rcode, &stat)
2429         //fmt.Printf("--D-- Client: %v (%v), rcode, stat)
2430         return rcode, ress
2431     }
2432
2433     <a name="remote-sh">Remote Shell</a>
2434     // gcp file [...] { [host]:[port]:[dir] | dir } // -p | -no-p
2435     func (gsh*GshContext)FileCopy(argv[string]){
2436         var host = ""
2437         var port = ""
2438         var upload = false
2439         var download = false
2440         var xargv = []string{"rex-gcp"}
2441         var srcv = []string{}
2442         var dstv = []string{}
2443         argv = argv[1:]
2444
2445         for _,v := range argv {
2446             /*
2447             if v[0] == '-' { // might be a pseudo file (generated date)
2448                 continue
2449             }
2450             */
2451             obj := strings.Split(v, ":")
2452             //fmt.Printf("%d %v %v\n", len(obj), v, obj)
2453             if 1 < len(obj) {
2454                 host = obj[0]
2455                 file := ""
2456                 if 0 < len(host) {
2457                     gsh.LastServer.host = host
2458                 } else {
2459                     host = gsh.LastServer.host
2460                     port = gsh.LastServer.port
2461                 }
2462                 if 2 < len(obj) {
2463                     port = obj[1]
2464                     if 0 < len(port) {
2465                         gsh.LastServer.port = port
2466                     } else {
2467                         port = gsh.LastServer.port
2468                     }
2469                     file = obj[2]
2470                 } else {
2471                     file = obj[1]
2472                 }
2473                 if len(srcv) == 0 {
2474                     download = true
2475                     srcv = append(srcv, file)
2476                     continue
2477                 }
2478                 upload = true
2479                 dstv = append(dstv, file)
2480                 continue

```

```

2481     }
2482     /*
2483     idx := strings.Index(v,":")
2484     if 0 <= idx {
2485         remote = v[0:idx]
2486         if len(srcv) == 0 {
2487             download = true
2488             srcv = append(srcv,v[idx+1:])
2489             continue
2490         }
2491         upload = true
2492         dstv = append(dstv,v[idx+1:])
2493         continue
2494     }
2495     */
2496     if download {
2497         dstv = append(dstv,v)
2498     }else{
2499         srcv = append(srcv,v)
2500     }
2501 }
2502 hostport := "@" + host + ":" + port
2503 if upload {
2504     if host != "" { xargv = append(xargv,hostport) }
2505     xargv = append(xargv,"PUT")
2506     xargv = append(xargv,srcv[0]...)
2507     xargv = append(xargv,dstv[0]...)
2508     //fmt.Printf("--I-- FileCopy PUT gsh://%/&v < %v // %v\n",hostport,dstv,srcv,xargv)
2509     fmt.Printf("--I-- FileCopy PUT gsh://%/&v < %v\n",hostport,dstv,srcv)
2510     gsh.RexecClient(xargv)
2511 }else
2512 if download {
2513     if host != "" { xargv = append(xargv,hostport) }
2514     xargv = append(xargv,"GET")
2515     xargv = append(xargv,srcv[0]...)
2516     xargv = append(xargv,dstv[0]...)
2517     //fmt.Printf("--I-- FileCopy GET gsh://%/&v/&v > %v // %v\n",hostport,srcv,dstv,xargv)
2518     fmt.Printf("--I-- FileCopy GET gsh://%/&v/&v > %v\n",hostport,srcv,dstv)
2519     gsh.RexecClient(xargv)
2520 }else{
2521 }
2522 }
2523
2524 // target
2525 func (gsh*GshContext)Trelpath(rloc string)(string){
2526     cwd, _ := os.Getwd()
2527     os.Chdir(gsh.RWD)
2528     os.Chdir(rloc)
2529     twd, _ := os.Getwd()
2530     os.Chdir(cwd)
2531
2532     tpath := twd + "/" + rloc
2533     return tpath
2534 }
2535 // join to remote GShell - [user@]host[:port] or cd host[:port]:path
2536 func (gsh*GshContext)Rjoin(argv[]string){
2537     if len(argv) <= 1 {
2538         fmt.Printf("--I-- current server = %v\n",gsh.RSERV)
2539         return
2540     }
2541     serv := argv[1]
2542     servv := strings.Split(serv,":")
2543     if 1 <= len(servv) {
2544         if servv[0] == "lo" {
2545             servv[0] = "localhost"
2546         }
2547     }
2548     switch len(servv) {
2549     case 1:
2550         //if strings.Index(serv,":") < 0 {
2551             serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
2552         //}
2553     case 2: // host:port
2554         serv = strings.Join(servv,":")
2555     }
2556     xargv := []string{"rex-join","@"+serv,"HELO"}
2557     rcode,stat := gsh.RexecClient(xargv)
2558     if (rcode / 100) == 2 {
2559         fmt.Printf("--I-- OK Joined (%v) %v\n",rcode,stat)
2560         gsh.RSERV = serv
2561     }else{
2562         fmt.Printf("--I-- NG, could not joined (%v) %v\n",rcode,stat)
2563     }
2564 }
2565 func (gsh*GshContext)Rexec(argv[]string){
2566     if len(argv) <= 1 {
2567         fmt.Printf("--I-- rexec command [ | {file || {command} ]\n",gsh.RSERV)
2568         return
2569     }
2570     /*
2571     nargv := gshScanArg(strings.Join(argv," "),0)
2572     fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
2573     if nargv[1][0] != '{' {
2574         nargv[1] = "{" + nargv[1] + "}"
2575         fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
2576     }
2577     argv = nargv
2578     */
2579     nargv := []string{}
2580     nargv = append(nargv,"{"+strings.Join(argv[1:]," ")+"}")
2581     fmt.Printf("--D-- nargc=%d %v\n",len(nargv),nargv)
2582     argv = nargv
2583
2584     xargv := []string{"rex-exec","@"+gsh.RSERV,"GET"}
2585     xargv = append(xargv,argv...)
2586     xargv = append(xargv,"/dev/tty")
2587     rcode,stat := gsh.RexecClient(xargv)
2588     if (rcode / 100) == 2 {
2589         fmt.Printf("--I-- OK Rexec (%v) %v\n",rcode,stat)
2590     }else{
2591         fmt.Printf("--I-- NG Rexec (%v) %v\n",rcode,stat)
2592     }
2593 }
2594 }
2595 func (gsh*GshContext)Rchdir(argv[]string){
2596     if len(argv) <= 1 {
2597         return
2598     }
2599     cwd, _ := os.Getwd()
2600     os.Chdir(gsh.RWD)
2601     os.Chdir(argv[1])
2602     twd, _ := os.Getwd()
2603     gsh.RWD = twd
2604     fmt.Printf("--I-- JWD=%v\n",twd)

```

```

2605     os.Chdir(cwd)
2606 }
2607 func (gsh*GshContext)Rpwd(argv[]string){
2608     fmt.Printf("%v\n",gsh.RWD)
2609 }
2610 func (gsh*GshContext)Rls(argv[]string){
2611     cwd, _ := os.Getwd()
2612     os.Chdir(gsh.RWD)
2613     argv[0] = "-ls"
2614     gsh.xFind(argv)
2615     os.Chdir(cwd)
2616 }
2617 func (gsh*GshContext)Rput(argv[]string){
2618     var local string = ""
2619     var remote string = ""
2620     if 1 < len(argv) {
2621         local = argv[1]
2622         remote = local // base name
2623     }
2624     if 2 < len(argv) {
2625         remote = argv[2]
2626     }
2627     fmt.Printf("--I-- jput from=%v to=%v\n",local,gsh.Trelpath(remote))
2628 }
2629 func (gsh*GshContext)Rget(argv[]string){
2630     var remote string = ""
2631     var local string = ""
2632     if 1 < len(argv) {
2633         remote = argv[1]
2634         local = remote // base name
2635     }
2636     if 2 < len(argv) {
2637         local = argv[2]
2638     }
2639     fmt.Printf("--I-- jget from=%v to=%v\n",gsh.Trelpath(remote),local)
2640 }
2641 }
2642 // <a name="network">network</a>
2643 // -s, -si, -so // bi-directional, source, sync (maybe socket)
2644 func (gshCtx*GshContext)sconnect(inTCP bool, argv []string) {
2645     gshPA := gshCtx.gshPA
2646     if len(argv) < 2 {
2647         fmt.Printf("Usage: -s [host]:[port[.udp]]\n")
2648         return
2649     }
2650     remote := argv[1]
2651     if remote == ":" { remote = "0.0.0.0:9999" }
2652
2653     if inTCP { // TCP
2654         dport, err := net.ResolveTCPAddr("tcp",remote);
2655         if err != nil {
2656             fmt.Printf("Address error: %s (%s)\n",remote,err)
2657             return
2658         }
2659         conn, err := net.DialTCP("tcp",nil,dport)
2660         if err != nil {
2661             fmt.Printf("Connection error: %s (%s)\n",remote,err)
2662             return
2663         }
2664         file, _ := conn.File();
2665         fd := file.Fd()
2666         fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,fd)
2667
2668         savfd := gshPA.Files[1]
2669         gshPA.Files[1] = fd;
2670         gshCtx.gshellv(argv[2:])
2671         gshPA.Files[1] = savfd
2672         file.Close()
2673         conn.Close()
2674     }else{
2675         //dport, err := net.ResolveUDPAddr("udp4",remote);
2676         dport, err := net.ResolveUDPAddr("udp",remote);
2677         if err != nil {
2678             fmt.Printf("Address error: %s (%s)\n",remote,err)
2679             return
2680         }
2681         //conn, err := net.DialUDP("udp4",nil,dport)
2682         conn, err := net.DialUDP("udp",nil,dport)
2683         if err != nil {
2684             fmt.Printf("Connection error: %s (%s)\n",remote,err)
2685             return
2686         }
2687         file, _ := conn.File();
2688         fd := file.Fd()
2689
2690         ar := conn.RemoteAddr()
2691         //al := conn.LocalAddr()
2692         fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
2693             remote,ar.String(),fd)
2694
2695         savfd := gshPA.Files[1]
2696         gshPA.Files[1] = fd;
2697         gshCtx.gshellv(argv[2:])
2698         gshPA.Files[1] = savfd
2699         file.Close()
2700         conn.Close()
2701     }
2702 }
2703 func (gshCtx*GshContext)saccept(inTCP bool, argv []string) {
2704     gshPA := gshCtx.gshPA
2705     if len(argv) < 2 {
2706         fmt.Printf("Usage: -ac [host]:[port[.udp]]\n")
2707         return
2708     }
2709     local := argv[1]
2710     if local == ":" { local = "0.0.0.0:9999" }
2711     if inTCP { // TCP
2712         port, err := net.ResolveTCPAddr("tcp",local);
2713         if err != nil {
2714             fmt.Printf("Address error: %s (%s)\n",local,err)
2715             return
2716         }
2717         //fmt.Printf("Listen at %s...\n",local);
2718         sconn, err := net.ListenTCP("tcp", port)
2719         if err != nil {
2720             fmt.Printf("Listen error: %s (%s)\n",local,err)
2721             return
2722         }
2723         //fmt.Printf("Accepting at %s...\n",local);
2724         aconn, err := sconn.AcceptTCP()
2725         if err != nil {
2726             fmt.Printf("Accept error: %s (%s)\n",local,err)
2727             return
2728         }

```

```

2729     file, _ := aconn.File()
2730     fd := file.Fd()
2731     fmt.Printf("Accepted TCP at %s [%d]\n", local, fd)
2732
2733     savfd := gshPA.Files[0]
2734     gshPA.Files[0] = fd;
2735     gshCtx.gshelv(argv[2:])
2736     gshPA.Files[0] = savfd
2737
2738     sconn.Close();
2739     aconn.Close();
2740     file.Close();
2741 }else{
2742 //port, err := net.ResolveUDPAddr("udp4", local);
2743 port, err := net.ResolveUDPAddr("udp", local);
2744 if err != nil {
2745     fmt.Printf("Address error: %s (%s)\n", local, err)
2746     return
2747 }
2748 fmt.Printf("Listen UDP at %s...\n", local);
2749 //uconn, err := net.ListenUDP("udp4", port)
2750 uconn, err := net.ListenUDP("udp", port)
2751 if err != nil {
2752     fmt.Printf("Listen error: %s (%s)\n", local, err)
2753     return
2754 }
2755 file, _ := uconn.File()
2756 fd := file.Fd()
2757 ar := uconn.RemoteAddr()
2758 remote := ""
2759 if ar != nil { remote = ar.String() }
2760 if remote == "" { remote = "?" }
2761
2762 // not yet received
2763 //fmt.Printf("Accepted at %s [%d] <- %s\n", local, fd, "")
2764
2765 savfd := gshPA.Files[0]
2766 gshPA.Files[0] = fd;
2767 savenv := gshPA.Env
2768 gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
2769 gshCtx.gshelv(argv[2:])
2770 gshPA.Env = savenv
2771 gshPA.Files[0] = savfd
2772
2773 uconn.Close();
2774 file.Close();
2775 }
2776 }
2777
2778 // empty line command
2779 func (gshCtx*GshContext)xPwd(argv []string){
2780 // execute context command, pwd + date
2781 // context notation, representation scheme, to be resumed at re-login
2782 cwd, _ := os.Getwd()
2783
2784 switch {
2785 case isin("-a", argv):
2786     gshCtx.ShowChdirHistory(argv)
2787 case isin("-ls", argv):
2788     showFileInfo(cwd, argv)
2789 default:
2790     fmt.Printf("%s\n", cwd)
2791 case isin("-v", argv): // obsolete empty command
2792     t := time.Now()
2793     date := t.Format(time.UnixDate)
2794     exe, _ := os.Executable()
2795     host, _ := os.Hostname()
2796     fmt.Printf("PWD=\"%s\", cwd)
2797     fmt.Printf(" HOST=\"%s\", host)
2798     fmt.Printf(" DATE=\"%s\", date)
2799     fmt.Printf(" TIME=\"%s\", t.String())
2800     fmt.Printf(" PID=\"%d\", os.Getpid())
2801     fmt.Printf(" EXE=\"%s\", exe)
2802     fmt.Printf("\n")
2803 }
2804
2805 // <a name="history">History</a>
2806 // these should be browsed and edited by HTTP browser
2807 // show the time of command with -t and direcotry with -ls
2808 // openfile-history, sort by -a -m -c
2809 // sort by elapsed time by -t -s
2810 // search by "more" like interface
2811 // edit history
2812 // sort history, and wc or uniq
2813 // CPU and other resource consumptions
2814 // limit showing range (by time or so)
2815 // export / import history
2816 func (gshCtx *GshContext)xHistory(argv []string){
2817     atWorkDirX := -1
2818     if 1 < len(argv) && strBegins(argv[1], "0") {
2819         atWorkDirX, _ = strconv.Atoi(argv[1][1:])
2820     }
2821     //fmt.Printf("--D-- showHistory(%v)\n", argv)
2822     for i, v := range gshCtx.CommandHistory {
2823         // exclude commands not to be listed by default
2824         // internal commands may be suppressed by default
2825         if v.CmdLine == "" && !isin("-a", argv) {
2826             continue;
2827         }
2828         if 0 <= atWorkDirX {
2829             if v.WorkDirX != atWorkDirX {
2830                 continue
2831             }
2832         }
2833         if !isin("-n", argv){ // like "fc"
2834             fmt.Printf("!%d ", i)
2835         }
2836         if isin("-v", argv){
2837             fmt.Println(v) // should be with it date
2838         }else{
2839             if isin("-l", argv) || isin("-l0", argv) {
2840                 elps := v.EndAt.Sub(v.StartAt);
2841                 start := v.StartAt.Format(time.Stamp)
2842                 fmt.Printf("%d ", v.WorkDirX)
2843                 fmt.Printf("[%v] %11v/t ", start, elps)
2844             }
2845             if isin("-l", argv) && !isin("-l0", argv){
2846                 fmt.Printf("%v", Rusagef("%t %u\t// %s", argv, v.Rusagev))
2847             }
2848             if isin("-at", argv) { // isin("-ls", argv){
2849                 dhi := v.WorkDirX // workdir history index
2850                 fmt.Printf("@%d %s\t", dhi, v.WorkDir)
2851                 // show the FileInfo of the output command??
2852             }

```

```

2853         fmt.Printf("%s",v.CmdLine)
2854         fmt.Printf("\n")
2855     }
2856 }
2857 }
2858 // !n - history index
2859 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
2860     if gline[0] == '!' {
2861         hix, err := strconv.Atoi(gline[1:])
2862         if err != nil {
2863             fmt.Printf("--E-- (%s : range)\n",hix)
2864             return "", false, true
2865         }
2866         if hix < 0 || len(gshCtx.CommandHistory) <= hix {
2867             fmt.Printf("--E-- (%d : out of range)\n",hix)
2868             return "", false, true
2869         }
2870         return gshCtx.CommandHistory[hix].CmdLine, false, false
2871     }
2872     // search
2873     //for i, v := range gshCtx.CommandHistory {
2874     //}
2875     return gline, false, false
2876 }
2877 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
2878     if 0 <= hix && hix < len(gsh.CommandHistory) {
2879         return gsh.CommandHistory[hix].CmdLine,true
2880     }
2881     return "",false
2882 }
2883
2884 // temporary adding to PATH environment
2885 // cd name -lib for LD_LIBRARY_PATH
2886 // chdir with directory history (date + full-path)
2887 // -s for sort option (by visit date or so)
2888 func (gsh*GshContext)ShowChdirHistory1(i int,v GChdirHistory, argv []string){
2889     fmt.Printf(":%-2d ",v.CmdIndex) // the first command at this WorkDir
2890     fmt.Printf("%s",i)
2891     fmt.Printf("[%v] ",v.MovedAt.Format(time.Stamp))
2892     showFileInfo(v.Dir,argv)
2893 }
2894 func (gsh*GshContext)ShowChdirHistory(argv []string){
2895     for i, v := range gsh.ChdirHistory {
2896         gsh.ShowChdirHistory1(i,v,argv)
2897     }
2898 }
2899 func skipOpts(argv[]string)(int){
2900     for i,v := range argv {
2901         if strBegins(v,"-") {
2902             }else{
2903                 return i
2904             }
2905     }
2906     return -1
2907 }
2908 func (gshCtx*GshContext)xChdir(argv []string){
2909     cdhist := gshCtx.ChdirHistory
2910     if isin("?",argv) || isin("-t",argv) || isin("-a",argv) {
2911         gshCtx.ShowChdirHistory(argv)
2912         return
2913     }
2914     pwd, _ := os.Getwd()
2915     dir := ""
2916     if len(argv) <= 1 {
2917         dir = toFullPath("--")
2918     }else{
2919         i := skipOpts(argv[1:])
2920         if i < 0 {
2921             dir = toFullPath("--")
2922         }else{
2923             dir = argv[1+i]
2924         }
2925     }
2926     if strBegins(dir,"@") {
2927         if dir == "@0" { // obsolete
2928             dir = gshCtx.StartDir
2929         }else
2930         if dir == "@!" {
2931             index := len(cdhist) - 1
2932             if 0 < index { index -= 1 }
2933             dir = cdhist[index].Dir
2934         }else{
2935             index, err := strconv.Atoi(dir[1:])
2936             if err != nil {
2937                 fmt.Printf("--E-- xChdir(%v)\n",err)
2938                 dir = "?"
2939             }else
2940             if len(gshCtx.ChdirHistory) <= index {
2941                 fmt.Printf("--E-- xChdir(history range error)\n")
2942                 dir = "?"
2943             }else{
2944                 dir = cdhist[index].Dir
2945             }
2946         }
2947     }
2948     if dir != "?" {
2949         err := os.Chdir(dir)
2950         if err != nil {
2951             fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
2952         }else{
2953             cwd, _ := os.Getwd()
2954             if cwd != pwd {
2955                 hist1 := GChdirHistory { }
2956                 hist1.Dir = cwd
2957                 hist1.MovedAt = time.Now()
2958                 hist1.CmdIndex = len(gshCtx.CommandHistory)+1
2959                 gshCtx.ChdirHistory = append(cdhist,hist1)
2960                 if !isin("-s",argv){
2961                     //cwd, _ := os.Getwd()
2962                     //fmt.Printf("%s\n",cwd)
2963                     ix := len(gshCtx.ChdirHistory)-1
2964                     gshCtx.ShowChdirHistory1(ix,hist1,argv)
2965                 }
2966             }
2967         }
2968     }
2969     if isin("-ls",argv){
2970         cwd, _ := os.Getwd()
2971         showFileInfo(cwd,argv);
2972     }
2973 }
2974 func TimeValSub(tv1 *syscall.Timeval, tv2 *syscall.Timeval){
2975     *tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
2976 }

```



```

2977 func RusageSubv(ru1, ru2 [2]syscall.Rusage)([2]syscall.Rusage){
2978     TimeValSub(&ru1[0].Utime,&ru2[0].Utime)
2979     TimeValSub(&ru1[0].Stime,&ru2[0].Stime)
2980     TimeValSub(&ru1[1].Utime,&ru2[1].Utime)
2981     TimeValSub(&ru1[1].Stime,&ru2[1].Stime)
2982     return ru1
2983 }
2984 func TimeValAdd(tv1 syscall.Timeval, tv2 syscall.Timeval)(syscall.Timeval){
2985     tvs := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
2986     return tvs
2987 }
2988 /*
2989 func RusageAddv(ru1, ru2 [2]syscall.Rusage)([2]syscall.Rusage){
2990     TimeValAdd(ru1[0].Utime,ru2[0].Utime)
2991     TimeValAdd(ru1[0].Stime,ru2[0].Stime)
2992     TimeValAdd(ru1[1].Utime,ru2[1].Utime)
2993     TimeValAdd(ru1[1].Stime,ru2[1].Stime)
2994     return ru1
2995 }
2996 */
2997
2998 // <a name="rusage">Resource Usage</a>
2999 func sRusagef(fmtspec string, argv []string, ru [2]syscall.Rusage)(string){
3000     // ru[0] self , ru[1] children
3001     ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3002     st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3003     uu := (ut.Sec*1000000 + int64(ut.Usec)) * 1000
3004     su := (st.Sec*1000000 + int64(st.Usec)) * 1000
3005     tu := uu + su
3006     ret := fmt.Sprintf("%v/sum",abftime(tu))
3007     ret += fmt.Sprintf(", %v/usr",abftime(uu))
3008     ret += fmt.Sprintf(", %v/sys",abftime(su))
3009     return ret
3010 }
3011 func Rusagef(fmtspec string, argv []string, ru [2]syscall.Rusage)(string){
3012     ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3013     st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3014     fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
3015     fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
3016     return ""
3017 }
3018 func Getrusagev()([2]syscall.Rusage){
3019     var ruv = [2]syscall.Rusage{}
3020     syscall.Getrusage(syscall.RUSAGE_SELF,&ruv[0])
3021     syscall.Getrusage(syscall.RUSAGE_CHILDREN,&ruv[1])
3022     return ruv
3023 }
3024 func showRusage(what string,argv []string, ru *syscall.Rusage){
3025     fmt.Printf("%s: ",what);
3026     fmt.Printf("Utr=%d.%06ds",ru.Utime.Sec,ru.Utime.Usec)
3027     fmt.Printf(" Sys=%d.%06ds",ru.Stime.Sec,ru.Stime.Usec)
3028     fmt.Printf(" Rss=%vB",ru.Maxrss)
3029     if isin("-l",argv) {
3030         fmt.Printf(" MinFlt=%v",ru.Minflt)
3031         fmt.Printf(" MajFlt=%v",ru.Majflt)
3032         fmt.Printf(" IdRSS=%vB",ru.Idrss)
3033         fmt.Printf(" IdRSS=%vB",ru.Idrss)
3034         fmt.Printf(" Nswap=%vB",ru.Nswap)
3035         fmt.Printf(" Read=%v",ru.Inblock)
3036         fmt.Printf(" Write=%v",ru.Oublock)
3037     }
3038     fmt.Printf(" Snd=%v",ru.Msgsnd)
3039     fmt.Printf(" Rcv=%v",ru.Msgrcv)
3040     //if isin("-l",argv) {
3041         fmt.Printf(" Sig=%v",ru.Nsignals)
3042     //}
3043     fmt.Printf("\n");
3044 }
3045 func (gshCtx *GshContext)xTime(argv[]string)(bool){
3046     if 2 <= len(argv){
3047         gshCtx.LastRusage = syscall.Rusage{}
3048         rusagev1 := Getrusagev()
3049         fin := gshCtx.gshellyv(argv[1:])
3050         rusagev2 := Getrusagev()
3051         showRusage(argv[1],argv,&gshCtx.LastRusage)
3052         rusagev := RusageSubv(rusagev2,rusagev1)
3053         showRusage("self",argv,&rusagev[0])
3054         showRusage("chld",argv,&rusagev[1])
3055         return fin
3056     }else{
3057         rusage:= syscall.Rusage {}
3058         syscall.Getrusage(syscall.RUSAGE_SELF,&rusage)
3059         showRusage("self",argv, &rusage)
3060         syscall.Getrusage(syscall.RUSAGE_CHILDREN,&rusage)
3061         showRusage("chld",argv, &rusage)
3062         return false
3063     }
3064 }
3065 func (gshCtx *GshContext)xJobs(argv[]string){
3066     fmt.Printf("%d Jobs",len(gshCtx.BackGroundJobs))
3067     for ji, pid := range gshCtx.BackGroundJobs {
3068         //wstat := syscall.WaitStatus {0}
3069         rusage := syscall.Rusage {}
3070         //wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&rusage);
3071         wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
3072         if err != nil {
3073             fmt.Printf("--E-- %%%d [%d] (%v)\n",ji,pid,err)
3074         }else{
3075             fmt.Printf("%%d[%d] (%d)\n",ji,pid,wpid)
3076             showRusage("chld",argv,&rusage)
3077         }
3078     }
3079 }
3080 func (gsh*GshContext)inBackground(argv[]string)(bool){
3081     if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n",argv) }
3082     gsh.BackGround = true // set background option
3083     xfin := false
3084     xfin = gsh.gshellyv(argv)
3085     gsh.BackGround = false
3086     return xfin
3087 }
3088 // -o file without command means just opening it and refer by #N
3089 // should be listed by "files" command
3090 func (gshCtx*GshContext)xOpen(argv[]string){
3091     var pv = []int{-1,-1}
3092     err := syscall.Pipe(pv)
3093     fmt.Printf("--I-- pipe(=[#&d,#&d] (%v)\n",pv[0],pv[1],err)
3094 }
3095 func (gshCtx*GshContext)fromPipe(argv[]string){
3096 }
3097 func (gshCtx*GshContext)xClose(argv[]string){
3098 }
3099
3100 // <a name="redirect">redirect</a>

```

```

3101 func (gshCtx *GshContext) redirect(argv []string) (bool) {
3102     if len(argv) < 2 {
3103         return false
3104     }
3105 }
3106 cmd := argv[0]
3107 fname := argv[1]
3108 var file *os.File = nil
3109
3110 fdix := 0
3111 mode := os.O_RDONLY
3112
3113 switch {
3114 case cmd == "-i" || cmd == "<":
3115     fdix = 0
3116     mode = os.O_RDONLY
3117 case cmd == "-o" || cmd == ">":
3118     fdix = 1
3119     mode = os.O_RDWR | os.O_CREATE
3120 case cmd == "-a" || cmd == ">>":
3121     fdix = 1
3122     mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3123 }
3124 if fname[0] == '#' {
3125     fd, err := strconv.Atoi(fname[1:])
3126     if err != nil {
3127         fmt.Printf("--E-- (%v)\n", err)
3128         return false
3129     }
3130     file = os.NewFile(uintptr(fd), "MaybePipe")
3131 } else {
3132     xfile, err := os.OpenFile(argv[1], mode, 0600)
3133     if err != nil {
3134         fmt.Printf("--E-- (%s)\n", err)
3135         return false
3136     }
3137     file = xfile
3138 }
3139 gshPA := gshCtx.gshPA
3140 savfd := gshPA.Files[fdix]
3141 gshPA.Files[fdix] = file.Fd()
3142 fmt.Printf("--I-- Opened [%d] %s\n", file.Fd(), argv[1])
3143 gshCtx.gshellv(argv[2:])
3144 gshPA.Files[fdix] = savfd
3145
3146 return false
3147 }
3148
3149 //fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
3150 func httpHandler(res http.ResponseWriter, req *http.Request) {
3151     path := req.URL.Path
3152     fmt.Printf("--I-- Got HTTP Request [%s]\n", path)
3153     {
3154         gshCtxBuf, _ := setupGshContext()
3155         gshCtx := *gshCtxBuf
3156         fmt.Printf("--I-- %s\n", path[1:])
3157         gshCtx.tgshell1(path[1:])
3158     }
3159     fmt.Fprintf(res, "Hello(^-^)/\n%s\n", path)
3160 }
3161 func (gshCtx *GshContext) httpServer(argv []string) {
3162     http.HandleFunc("/", httpHandler)
3163     accport := "localhost:9999"
3164     fmt.Printf("--I-- HTTP Server Start at [%s]\n", accport)
3165     http.ListenAndServe(accport, nil)
3166 }
3167 func (gshCtx *GshContext) xGo(argv []string) {
3168     go gshCtx.gshellv(argv[1:])
3169 }
3170 func (gshCtx *GshContext) xPs(argv []string) () {
3171 }
3172
3173 // <a name="plugin">Plugin</a>
3174 // plugin [-ls [names]] to list plugins
3175 // Reference: <a href="https://golang.org/src/plugin/">plugin</a> source code
3176 func (gshCtx *GshContext) whichPlugin(name string, argv []string) (pi *PluginInfo) {
3177     pi = nil
3178     for _, p := range gshCtx.PluginFuncs {
3179         if p.Name == name && pi == nil {
3180             pi = &p
3181         }
3182         if !isin("-s", argv) {
3183             //fmt.Printf("%v %v ", i, p)
3184             if isin("-ls", argv) {
3185                 showFileInfo(p.Path, argv)
3186             } else {
3187                 fmt.Printf("%s\n", p.Name)
3188             }
3189         }
3190     }
3191     return pi
3192 }
3193 func (gshCtx *GshContext) xPlugin(argv []string) (error) {
3194     if len(argv) == 0 || argv[0] == "-ls" {
3195         gshCtx.whichPlugin("", argv)
3196         return nil
3197     }
3198     name := argv[0]
3199     pin := gshCtx.whichPlugin(name, []string{"-s"})
3200     if pin != nil {
3201         os.Args = argv // should be recovered?
3202         pin.Addr.(func())()
3203         return nil
3204     }
3205     sofile := toFullpath(argv[0] + ".so") // or find it by which($PATH)
3206
3207     p, err := plugin.Open(sofile)
3208     if err != nil {
3209         fmt.Printf("--E-- plugin.Open(%s)(%v)\n", sofile, err)
3210         return err
3211     }
3212     fname := "Main"
3213     f, err := p.Lookup(fname)
3214     if( err != nil ){
3215         fmt.Printf("--E-- plugin.Lookup(%s)(%v)\n", fname, err)
3216         return err
3217     }
3218     pin := PluginInfo { p, f, name, sofile }
3219     gshCtx.PluginFuncs = append(gshCtx.PluginFuncs, pin)
3220     fmt.Printf("--I-- added (%d)\n", len(gshCtx.PluginFuncs))
3221
3222     //fmt.Printf("--I-- first call(%s:%s)%v\n", sofile, fname, argv)
3223     os.Args = argv
3224     f.(func())()

```

```

3225     return err
3226 }
3227 func (gshCtx *GshContext) Args(argv []string) {
3228     for i, v := range os.Args {
3229         fmt.Printf("[%v] %v\n", i, v)
3230     }
3231 }
3232 func (gshCtx *GshContext) showVersion(argv []string) {
3233     if isin("-l", argv) {
3234         fmt.Printf("%v/%v (%v)", NAME, VERSION, DATE);
3235     } else {
3236         fmt.Printf("%v", VERSION);
3237     }
3238     if isin("-a", argv) {
3239         fmt.Printf(" %s", AUTHOR)
3240     }
3241     if !isin("-n", argv) {
3242         fmt.Printf("\n")
3243     }
3244 }
3245 }
3246 // <a name="scanf">Scanf</a> // string decomposer
3247 // scanf [format] [input]
3248 func scanv(sstr string)(strv []string){
3249     strv = strings.Split(sstr, " ")
3250     return strv
3251 }
3252 func scanUntil(src, end string)(rstr string, leng int){
3253     idx := strings.Index(src, end)
3254     if 0 <= idx {
3255         rstr = src[0:idx]
3256         return rstr, idx+leng(end)
3257     }
3258     return src, 0
3259 }
3260 }
3261 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
3262 func (gsh *GshContext) printVal(fmts string, vstr string, optv []string) {
3263     //fmt, err := strconv.Atoi(vstr)
3264     var ival int64 = 0
3265     n := 0
3266     err := error(nil)
3267     if strBegins(vstr, ".") {
3268         vx, _ := strconv.Atoi(vstr[1:])
3269         if vx < len(gsh.iValues) {
3270             vstr = gsh.iValues[vx]
3271         } else {
3272             }
3273     }
3274     // should use Eval()
3275     if strBegins(vstr, "0x") {
3276         n, err = fmt.Sscanf(vstr[2:], "%x", &ival)
3277     } else {
3278         n, err = fmt.Sscanf(vstr, "%d", &ival)
3279     }
3280     //fmt.Printf("--D-- n=%d err=(%v) (%s)=%v\n", n, err, vstr, ival)
3281     if n == 1 && err == nil {
3282         //fmt.Printf("--D-- formatn(%v) ival(%v)\n", fmts, ival)
3283         fmt.Printf("%"+fmts, ival)
3284     } else {
3285         if isin("-bn", optv) {
3286             fmt.Printf("%"+fmts, filepath.Base(vstr))
3287         } else {
3288             fmt.Printf("%"+fmts, vstr)
3289         }
3290     }
3291 }
3292 }
3293 func (gsh *GshContext) printfv(fmts, div string, argv []string, optv []string, list []string) {
3294     //fmt.Printf("%d", len(list))
3295     //curfmt := "v"
3296     outlen := 0
3297     curfmt := gsh.iFormat
3298     if 0 < len(fmts) {
3299         for xi := 0; xi < len(fmts); xi++ {
3300             fch := fmts[xi]
3301             if fch == '%' {
3302                 if xi+1 < len(fmts) {
3303                     curfmt = string(fmts[xi+1])
3304                 }
3305                 gsh.iFormat = curfmt
3306                 xi += 1
3307                 if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3308                     vals, leng := scanUntil(fmts[xi+2:], ")")
3309                     //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n", curfmt, vals, leng)
3310                     gsh.printVal(curfmt, vals, optv)
3311                     xi += 2+leng-1
3312                     outlen += 1
3313                 }
3314                 continue
3315             }
3316             if fch == '.' {
3317                 hi, leng := scanInt(fmts[xi+1:])
3318                 if 0 < leng {
3319                     if hi < len(gsh.iValues) {
3320                         gsh.printVal(curfmt, gsh.iValues[hi], optv)
3321                         outlen += 1 // should be the real length
3322                     } else {
3323                         fmt.Printf("(out-range)")
3324                     }
3325                     xi += leng
3326                     continue;
3327                 }
3328             }
3329             fmt.Printf("%c", fch)
3330             outlen += 1
3331         }
3332     } else {
3333         //fmt.Printf("--D-- print (%s)\n")
3334         for i, v := range list {
3335             if 0 < i {
3336                 fmt.Printf(div)
3337             }
3338             gsh.printVal(curfmt, v, optv)
3339             outlen += 1
3340         }
3341     }
3342     if 0 < outlen {
3343         fmt.Printf("\n")
3344     }
3345 }
3346 }
3347 func (gsh *GshContext) Scanv(argv []string) {
3348     //fmt.Printf("--D-- Scanv(%v)\n", argv)
3349     if len(argv) == 1 {

```

```

3349     return
3350 }
3351 argv = argv[1:]
3352 fmts := ""
3353 if strBegins(argv[0], "-F") {
3354     fmts = argv[0]
3355     gsh.iDelimiter = fmts
3356     argv = argv[1:]
3357 }
3358 input := strings.Join(argv, " ")
3359 if fmts == "" { // simple decomposition
3360     v := scanv(input)
3361     gsh.iValues = v
3362     //fmt.Printf("%v\n", strings.Join(v, ","))
3363 } else {
3364     v := make([]string, 8)
3365     n, err := fmt.Sscanf(input, fmts, &v[0], &v[1], &v[2], &v[3])
3366     fmt.Printf("--D-- Scanf ->(%v) n=%d err=(%v)\n", v, n, err)
3367     gsh.iValues = v
3368 }
3369 }
3370 func (gsh*GshContext)Printv(argv []string){
3371     if false { //88U
3372         fmt.Printf("%v\n", strings.Join(argv[1:], " "))
3373         return
3374     }
3375     //fmt.Printf("--D-- Printv(%v)\n", argv)
3376     //fmt.Printf("%v\n", strings.Join(gsh.iValues, ","))
3377     div := gsh.iDelimiter
3378     fmts := ""
3379     argv = argv[1:]
3380     if 0 < len(argv) {
3381         if strBegins(argv[0], "-F") {
3382             div = argv[0][2:]
3383             argv = argv[1:]
3384         }
3385     }
3386     optv := []string{}
3387     for _, v := range argv {
3388         if strBegins(v, "-"){
3389             optv = append(optv, v)
3390             argv = argv[1:]
3391         } else {
3392             break;
3393         }
3394     }
3395     if 0 < len(argv) {
3396         fmts = strings.Join(argv, " ")
3397     }
3398     gsh.printfv(fmts, div, argv, optv, gsh.iValues)
3399 }
3400 }
3401 func (gsh*GshContext)Basename(argv []string){
3402     for i, v := range gsh.iValues {
3403         gsh.iValues[i] = filepath.Base(v)
3404     }
3405 }
3406 func (gsh*GshContext)Sortv(argv []string){
3407     sv := gsh.iValues
3408     sort.Slice(sv, func(i, j int) bool {
3409         return sv[i] < sv[j]
3410     })
3411 }
3412 func (gsh*GshContext)Shiftv(argv []string){
3413     vi := len(gsh.iValues)
3414     if 0 < vi {
3415         if isin("-r", argv) {
3416             top := gsh.iValues[0]
3417             gsh.iValues = append(gsh.iValues[1:], top)
3418         } else {
3419             gsh.iValues = gsh.iValues[1:]
3420         }
3421     }
3422 }
3423 }
3424 func (gsh*GshContext)Enq(argv []string){
3425 }
3426 func (gsh*GshContext)Deq(argv []string){
3427 }
3428 func (gsh*GshContext)Push(argv []string){
3429     gsh.iValStack = append(gsh.iValStack, argv[1:])
3430     fmt.Printf("depth=%d\n", len(gsh.iValStack))
3431 }
3432 func (gsh*GshContext)Dump(argv []string){
3433     for i, v := range gsh.iValStack {
3434         fmt.Printf("%d %v\n", i, v)
3435     }
3436 }
3437 func (gsh*GshContext)Pop(argv []string){
3438     depth := len(gsh.iValStack)
3439     if 0 < depth {
3440         v := gsh.iValStack[depth-1]
3441         if isin("-cat", argv){
3442             gsh.iValues = append(gsh.iValues, v...)
3443         } else {
3444             gsh.iValues = v
3445         }
3446         gsh.iValStack = gsh.iValStack[0:depth-1]
3447         fmt.Printf("depth=%d %s\n", len(gsh.iValStack), gsh.iValues)
3448     } else {
3449         fmt.Printf("depth=%d\n", depth)
3450     }
3451 }
3452 }
3453 // <a name="interpreter">Command Interpreter</a>
3454 func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
3455     fin = false
3456 }
3457 if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr, "--I-- gshellv(%d)\n", len(argv)) }
3458 if len(argv) <= 0 {
3459     return false
3460 }
3461 xargv := []string{}
3462 for ai := 0; ai < len(argv); ai++ {
3463     xargv = append(xargv, strsubst(gshCtx, argv[ai], false))
3464 }
3465 argv = xargv
3466 if false {
3467     for ai := 0; ai < len(argv); ai++ {
3468         fmt.Printf("[%d] %s [%d]T\n",
3469             ai, argv[ai], len(argv[ai]), argv[ai])
3470     }
3471 }
3472 cmd := argv[0]

```

```

3473 if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr, "--I-- gshellv%d%\n", len(argv), argv) }
3474 switch { // https://tour.golang.org/flowcontrol/11
3475 case cmd == "":
3476     gshCtx.xPwd([]string{}); // empty command
3477 case cmd == "-x":
3478     gshCtx.CmdTrace = !gshCtx.CmdTrace
3479 case cmd == "-xt":
3480     gshCtx.CmdTime = !gshCtx.CmdTime
3481 case cmd == "-ot":
3482     gshCtx.sconnect(true, argv)
3483 case cmd == "-ou":
3484     gshCtx.sconnect(false, argv)
3485 case cmd == "-it":
3486     gshCtx.saccept(true, argv)
3487 case cmd == "-iu":
3488     gshCtx.saccept(false, argv)
3489 case cmd == "-i" || cmd == "<" || cmd == "-o" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
3490     gshCtx.redirect(argv)
3491 case cmd == "|":
3492     gshCtx.fromPipe(argv)
3493 case cmd == "args":
3494     gshCtx.Args(argv)
3495 case cmd == "-bg" || cmd == "-bg":
3496     rfin := gshCtx.inBackground(argv[1:])
3497     return rfin
3498 case cmd == "-bn":
3499     gshCtx.Basename(argv)
3500 case cmd == "call":
3501     _,_ = gshCtx.excommand(false, argv[1:])
3502 case cmd == "cd" || cmd == "chdir":
3503     gshCtx.xChdir(argv);
3504 case cmd == "-cksum":
3505     gshCtx.xFind(argv)
3506 case cmd == "-sum":
3507     gshCtx.xFind(argv)
3508 case cmd == "-sumtest":
3509     str := ""
3510     if 1 < len(argv) { str = argv[1] }
3511     crc := strCRC32(str, uint64(len(str)))
3512     fprintf(stderr, "%v %v\n", crc, len(str))
3513 case cmd == "close":
3514     gshCtx.xClose(argv)
3515 case cmd == "gcp":
3516     gshCtx.FileCopy(argv)
3517 case cmd == "dec" || cmd == "decode":
3518     gshCtx.Dec(argv)
3519 case cmd == "#define":
3520 case cmd == "dic" || cmd == "d":
3521     xDic(argv)
3522 case cmd == "dump":
3523     gshCtx.Dump(argv)
3524 case cmd == "echo" || cmd == "e":
3525     echo(argv, true)
3526 case cmd == "enc" || cmd == "encode":
3527     gshCtx.Enc(argv)
3528 case cmd == "env":
3529     env(argv)
3530 case cmd == "eval":
3531     xEval(argv[1:], true)
3532 case cmd == "ev" || cmd == "events":
3533     dumpEvents(argv)
3534 case cmd == "exec":
3535     _,_ = gshCtx.excommand(true, argv[1:])
3536     // should not return here
3537 case cmd == "exit" || cmd == "quit":
3538     // write Result code EXIT to 3>
3539     return true
3540 case cmd == "fdls":
3541     // dump the attributes of fds (of other process)
3542 case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
3543     gshCtx.xFind(argv[1:])
3544 case cmd == "fu":
3545     gshCtx.xFind(argv[1:])
3546 case cmd == "fork":
3547     // mainly for a server
3548 case cmd == "-gen":
3549     gshCtx.gen(argv)
3550 case cmd == "-go":
3551     gshCtx.xGo(argv)
3552 case cmd == "-grep":
3553     gshCtx.xFind(argv)
3554 case cmd == "gdeq":
3555     gshCtx.Deq(argv)
3556 case cmd == "genq":
3557     gshCtx.Enq(argv)
3558 case cmd == "gpop":
3559     gshCtx.Pop(argv)
3560 case cmd == "gpush":
3561     gshCtx.Push(argv)
3562 case cmd == "history" || cmd == "hi": // hi should be alias
3563     gshCtx.xHistory(argv)
3564 case cmd == "jobs":
3565     gshCtx.xJobs(argv)
3566 case cmd == "lnsp" || cmd == "nlsp":
3567     gshCtx.SplitLine(argv)
3568 case cmd == "-ls":
3569     gshCtx.xFind(argv)
3570 case cmd == "nop":
3571     // do nothing
3572 case cmd == "pipe":
3573     gshCtx.xOpen(argv)
3574 case cmd == "plug" || cmd == "plugin" || cmd == "pin":
3575     gshCtx.xPlugin(argv[1:])
3576 case cmd == "print" || cmd == "-pr":
3577     // output internal slice // also sprintf should be
3578     gshCtx.Printv(argv)
3579 case cmd == "ps":
3580     gshCtx.xPs(argv)
3581 case cmd == "pstitle":
3582     // to be gsh.title
3583 case cmd == "rexecd" || cmd == "rexd":
3584     gshCtx.RexecServer(argv)
3585 case cmd == "rexec" || cmd == "rex":
3586     gshCtx.RexecClient(argv)
3587 case cmd == "repeat" || cmd == "rep": // repeat cond command
3588     gshCtx.repeat(argv)
3589 case cmd == "replay":
3590     gshCtx.xReplay(argv)
3591 case cmd == "scan":
3592     // scan input (or so in fscanf) to internal slice (like Files or map)
3593     gshCtx.Scanv(argv)
3594 case cmd == "set":
3595     // set name ...
3596 case cmd == "serv":

```

```

3597     gshCtx.httpServer(argv)
3598 case cmd == "shift":
3599     gshCtx.Shiftv(argv)
3600 case cmd == "sleep":
3601     gshCtx.sleep(argv)
3602 case cmd == "-sort":
3603     gshCtx.Sortv(argv)
3604
3605 case cmd == "j" || cmd == "join":
3606     gshCtx.Rjoin(argv)
3607 case cmd == "a" || cmd == "alpa":
3608     gshCtx.Rexec(argv)
3609 case cmd == "jcd" || cmd == "jchdir":
3610     gshCtx.Rchdir(argv)
3611 case cmd == "jget":
3612     gshCtx.Rget(argv)
3613 case cmd == "jls":
3614     gshCtx.Rls(argv)
3615 case cmd == "jput":
3616     gshCtx.Rput(argv)
3617 case cmd == "jpwd":
3618     gshCtx.Rpwd(argv)
3619
3620 case cmd == "time":
3621     fin = gshCtx.xTime(argv)
3622 case cmd == "ungets":
3623     if l < len(argv) {
3624         ungets(argv[l]+"\\n")
3625     }else{
3626     }
3627 case cmd == "pwd":
3628     gshCtx.xPwd(argv);
3629 case cmd == "ver" || cmd == "-ver" || cmd == "version":
3630     gshCtx.showVersion(argv)
3631 case cmd == "where":
3632     // data file or so?
3633 case cmd == "which":
3634     which("PATH", argv);
3635 case cmd == "gj" && l < len(argv) && argv[l] == "listen":
3636     go gj_server(argv[l]);
3637 case cmd == "gj" && l < len(argv) && argv[l] == "serve":
3638     go gj_server(argv[l]);
3639 case cmd == "gj" && l < len(argv) && argv[l] == "join":
3640     go gj_client(argv[l]);
3641 case cmd == "gj":
3642     jsend(argv);
3643 case cmd == "jsend":
3644     jsend(argv);
3645 default:
3646     if gshCtx.whichPlugin(cmd,[]string{"-s"}) != nil {
3647         gshCtx.xPlugin(argv)
3648     }else{
3649         notfound,_ := gshCtx.excommand(false,argv)
3650         if notfound {
3651             fmt.Printf("--E-- command not found (%v)\\n",cmd)
3652         }
3653     }
3654 }
3655 return fin
3656 }
3657
3658 func (gsh*GshContext)gshelll(gline string) (rfin bool) {
3659     argv := strings.Split(string(gline)," ")
3660     fin := gsh.gshellv(argv)
3661     return fin
3662 }
3663 func (gsh*GshContext)tgshelll(gline string)(xfn bool){
3664     start := time.Now()
3665     fin := gsh.gshelll(gline)
3666     end := time.Now()
3667     elps := end.Sub(start);
3668     if gsh.CmdTime {
3669         fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + "(%d.%09ds)\\n",
3670             elps/1000000000,elps%1000000000)
3671     }
3672     return fin
3673 }
3674 func Ttyid() (int) {
3675     fi, err := os.Stdin.Stat()
3676     if err != nil {
3677         return 0;
3678     }
3679     //fmt.Printf("Stdin: %v Dev=%d\\n",
3680     // fi.Mode(),fi.Mode()&os.ModeDevice)
3681     if (fi.Mode() & os.ModeDevice) != 0 {
3682         stat := syscall.Stat_t{};
3683         err := syscall.Fstat(0,&stat)
3684         if err != nil {
3685             //fmt.Printf("--I-- Stdin: (%v)\\n",err)
3686         }else{
3687             //fmt.Printf("--I-- Stdin: rdev=%d %d\\n",
3688             // stat.Rdev&0xFF,stat.Rdev);
3689             //fmt.Printf("--I-- Stdin: tty%d\\n",stat.Rdev&0xFF);
3690             return int(stat.Rdev & 0xFF)
3691         }
3692     }
3693     return 0
3694 }
3695 func (gshCtx *GshContext) ttyfile() string {
3696     //fmt.Printf("--I-- GSH_HOME=%s\\n",gshCtx.GshHomeDir)
3697     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
3698         fmt.Sprintf("%02d",gshCtx.TerminalId)
3699     //strconv.Itoa(gshCtx.TerminalId)
3700     //fmt.Printf("--I-- ttyfile=%s\\n",ttyfile)
3701     return ttyfile
3702 }
3703 func (gshCtx *GshContext) ttyline>(*os.File){
3704     file, err := os.OpenFile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
3705     if err != nil {
3706         fmt.Printf("--F-- cannot open %s (%s)\\n",gshCtx.ttyfile(),err)
3707         return file;
3708     }
3709     return file
3710 }
3711 func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string) {
3712     if( skipping ){
3713         reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
3714         line,_,_ := reader.ReadLine()
3715         return string(line)
3716     }else
3717     if true {
3718         return xgetline(hix,prevline,gshCtx)
3719     }
3720     /*

```

```

3721     else
3722     if( with_exgetline && gshCtx.GetLine != "" ){
3723         //var xhix int64 = int64(hix); // cast
3724         newenv := os.Environ()
3725         newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )
3726     }
3727     tty := gshCtx.ttyline()
3728     tty.WriteString(prevline)
3729     Pa := os.ProcAttr {
3730         "", // start dir
3731         newenv, //os.Environ(),
3732         []*os.File{os.Stdin,os.Stdout,os.Stderr,tty},
3733         nil,
3734     }
3735     //fmt.Printf("--I-- getline=%s // %s\n",gsh_getlinev[0],gshCtx.GetLine)
3736     proc, err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&Pa)
3737     if err != nil {
3738         fmt.Printf("--F-- getline process error (%v)\n",err)
3739         // for ; ; { }
3740         return "exit (getline program failed)"
3741     }
3742     //stat, err := proc.Wait()
3743     proc.Wait()
3744     buff := make([]byte,LINESIZE)
3745     count, err := tty.Read(buff)
3746     //_, err = tty.Read(buff)
3747     //fmt.Printf("--D-- getline (%d)\n",count)
3748     if err != nil {
3749         if ! (count == 0) { // && err.String() == "EOF" } {
3750             fmt.Printf("--E-- getline error (%s)\n",err)
3751         }
3752     }else{
3753         //fmt.Printf("--I-- getline OK \"%s\"\n",buff)
3754     }
3755     tty.Close()
3756     gline := string(buff[0:count])
3757     return gline
3758 }else
3759 /*
3760 {
3761     // if isatty {
3762     fmt.Printf("!&d",hix)
3763     fmt.Print(PROMPT)
3764     // }
3765     reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
3766     line, _, _ := reader.ReadLine()
3767     return string(line)
3768 }
3769 }
3770
3771 //== begin ===== getline
3772 /*
3773 * getline.c
3774 * 2020-0819 extracted from dog.c
3775 * getline.go
3776 * 2020-0822 ported to Go
3777 */
3778 /*
3779 package main // getline main
3780 import (
3781     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
3782     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
3783     "os" // <a href="https://golang.org/pkg/os/">os</a>
3784     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
3785     // "bytes" // <a href="https://golang.org/pkg/bytes/">bytes</a>
3786     // "os/exec" // <a href="https://golang.org/pkg/os/exec/">os/exec</a>
3787 )
3788 */
3789
3790 // C language compatibility functions
3791 var errno = 0
3792 var stdin *os.File = os.Stdin
3793 var stdout *os.File = os.Stdout
3794 var stderr *os.File = os.Stderr
3795 var EOF = -1
3796 var NULL = 0
3797 type FILE os.File
3798 type StrBuff []byte
3799 var NULL_FP *os.File = nil
3800 var NULLSP = 0
3801 //var LINESIZE = 1024
3802
3803 func system(cmdstr string)(int){
3804     PA := syscall.ProcAttr {
3805         "", // the starting directory
3806         os.Environ(),
3807         []uintptr{os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()},
3808         nil,
3809     }
3810     argv := strings.Split(cmdstr, " ")
3811     pid,err := syscall.ForkExec(argv[0],argv,&PA)
3812     if( err != nil ){
3813         fmt.Printf("--E-- syscall(%v) err(%v)\n",cmdstr,err)
3814     }
3815     syscall.Wait4(pid,nil,0,nil)
3816
3817     /*
3818     argv := strings.Split(cmdstr, " ")
3819     fmt.Fprintf(os.Stderr,"--I-- system(%v)\n",argv)
3820     //cmd := exec.Command(argv[0]:...)
3821     cmd := exec.Command(argv[0],argv[1],argv[2])
3822     cmd.Stdin = strings.NewReader("output of system")
3823     var out bytes.Buffer
3824     cmd.Stdout = &out
3825     var serr bytes.Buffer
3826     cmd.Stderr = &serr
3827     err := cmd.Run()
3828     if err != nil {
3829         fmt.Fprintf(os.Stderr,"--E-- system(%v)err(%v)\n",argv,err)
3830         fmt.Printf("ERR:%s\n",serr.String())
3831     }else{
3832         fmt.Printf("%s",out.String())
3833     }
3834     */
3835     return 0
3836 }
3837 func atoi(str string)(ret int){
3838     ret,err := fmt.Sscanf(str,"%d",ret)
3839     if err == nil {
3840         return ret
3841     }else{
3842         // should set errno
3843         return 0
3844     }

```

```

3845 }
3846 func getenv(name string)(string){
3847     val, got := os.LookupEnv(name)
3848     if got {
3849         return val
3850     }else{
3851         return "?"
3852     }
3853 }
3854 func strcpy(dst StrBuff, src string){
3855     var i int
3856     srcb := []byte(src)
3857     for i = 0; i < len(src) && srcb[i] != 0; i++ {
3858         dst[i] = srcb[i]
3859     }
3860     dst[i] = 0
3861 }
3862 func xstrcpy(dst StrBuff, src StrBuff){
3863     dst = src
3864 }
3865 func strcat(dst StrBuff, src StrBuff){
3866     dst = append(dst,src...)
3867 }
3868 func strdup(str StrBuff)(string){
3869     return string(str[:strlen(str)])
3870 }
3871 func strlen(str string)(int){
3872     return len(str)
3873 }
3874 func strlen(str StrBuff)(int){
3875     var i int
3876     for i = 0; i < len(str) && str[i] != 0; i++ {
3877     }
3878     return i
3879 }
3880 func sizeof(data StrBuff)(int){
3881     return len(data)
3882 }
3883 func isatty(fd int)(ret int){
3884     return 1
3885 }
3886
3887 func fopen(file string,mode string)(fp*os.File){
3888     if mode == "r" {
3889         fp,err := os.Open(file)
3890         if( err != nil ){
3891             fmt.Printf("--E-- fopen(%s,%s)=(%v)\n",file,mode,err)
3892             return NULL_FP;
3893         }
3894         return fp;
3895     }else{
3896         fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
3897         if( err != nil ){
3898             return NULL_FP;
3899         }
3900         return fp;
3901     }
3902 }
3903 func fclose(fp*os.File){
3904     fp.Close()
3905 }
3906 func fflush(fp *os.File)(int){
3907     return 0
3908 }
3909 func fgetc(fp*os.File)(int){
3910     var buf [1]byte
3911     _,err := fp.Read(buf[0:1])
3912     if( err != nil ){
3913         return EOF;
3914     }else{
3915         return int(buf[0])
3916     }
3917 }
3918 func sfgets(str*string, size int, fp*os.File)(int){
3919     buf := make(StrBuff,size)
3920     var ch int
3921     var i int
3922     for i = 0; i < len(buf)-1; i++ {
3923         ch = fgetc(fp)
3924         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
3925         if( ch == EOF ){
3926             break;
3927         }
3928         buf[i] = byte(ch);
3929         if( ch == '\n' ){
3930             break;
3931         }
3932     }
3933     buf[i] = 0
3934     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
3935     return i
3936 }
3937 func fgets(buf StrBuff, size int, fp*os.File)(int){
3938     var ch int
3939     var i int
3940     for i = 0; i < len(buf)-1; i++ {
3941         ch = fgetc(fp)
3942         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
3943         if( ch == EOF ){
3944             break;
3945         }
3946         buf[i] = byte(ch);
3947         if( ch == '\n' ){
3948             break;
3949         }
3950     }
3951     buf[i] = 0
3952     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
3953     return i
3954 }
3955 func fputc(ch int , fp*os.File)(int){
3956     var buf [1]byte
3957     buf[0] = byte(ch)
3958     fp.Write(buf[0:1])
3959     return 0
3960 }
3961 func fputs(buf StrBuff, fp*os.File)(int){
3962     fp.Write(buf)
3963     return 0
3964 }
3965 func xfputss(str string, fp*os.File)(int){
3966     return fputs([]byte(str),fp)
3967 }
3968 func sscanf(str StrBuff,fmts string, params ...interface{})(int){

```



```

3969     fmt.Sprintf(string(str[0:strlen(str)]),fmts,params...)
3970     return 0
3971 }
3972 func fprintf(fp*os.File,fmts string, params ...interface{})(int){
3973     fmt.Fprintf(fp,fmts,params...)
3974     return 0
3975 }
3976
3977 // <a name="IME">Command Line IME</a>
3978 //----- MyIME
3979 var MyIMEVER = "MyIME/0.0.2";
3980 type RomKana struct {
3981     dic string // dictionary ID
3982     pat string // input pattern
3983     out string // output pattern
3984     hit int64 // count of hit and used
3985 }
3986 var dicents = 0
3987 var romkana [1024]RomKana
3988 var Romkan []RomKana
3989
3990 func isinDic(str string)(int){
3991     for i,v := range Romkan {
3992         if v.pat == str {
3993             return i
3994         }
3995     }
3996     return -1
3997 }
3998 const (
3999     DIC_COM_LOAD = "im"
4000     DIC_COM_DUMP = "s"
4001     DIC_COM_LIST = "ls"
4002     DIC_COM_ENA = "en"
4003     DIC_COM_DIS = "di"
4004 )
4005 func helpDic(argv []string){
4006     out := stderr
4007     cmd := ""
4008     if 0 < len(argv) { cmd = argv[0] }
4009     fprintf(out,"--- %v Usage\n",cmd)
4010     fprintf(out,"... Commands\n")
4011     fprintf(out,"... %v %3v [dicName] [dicURL] -- Import dictionary\n",cmd,DIC_COM_LOAD)
4012     fprintf(out,"... %v %3v [pattern] -- Search in dictionary\n",cmd,DIC_COM_DUMP)
4013     fprintf(out,"... %v %3v [dicName] -- List dictionaries\n",cmd,DIC_COM_LIST)
4014     fprintf(out,"... %v %3v [dicName] -- Disable dictionaries\n",cmd,DIC_COM_DIS)
4015     fprintf(out,"... %v %3v [dicName] -- Enable dictionaries\n",cmd,DIC_COM_ENA)
4016     fprintf(out,"... Keys ... %v\n","ESC can be used for '\\')
4017     fprintf(out,"... \\c -- Reverse the case of the last character\n",)
4018     fprintf(out,"... \\i -- Replace input with translated text\n",)
4019     fprintf(out,"... \\j -- On/Off translation mode\n",)
4020     fprintf(out,"... \\l -- Force Lower Case\n",)
4021     fprintf(out,"... \\u -- Force Upper Case (software CapsLock)\n",)
4022     fprintf(out,"... \\v -- Show translation actions\n",)
4023     fprintf(out,"... \\x -- Replace the last input character with it Hexa-Decimal\n",)
4024 }
4025 func xDic(argv[]string){
4026     if len(argv) <= 1 {
4027         helpDic(argv)
4028         return
4029     }
4030     argv = argv[1:]
4031     var debug = false
4032     var info = false
4033     var silent = false
4034     var dump = false
4035     var builtin = false
4036     cmd := argv[0]
4037     argv = argv[1:]
4038     opt := ""
4039     arg := ""
4040
4041     if 0 < len(argv) {
4042         arg1 := argv[0]
4043         if arg1[0] == '-' {
4044             switch arg1 {
4045                 default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
4046                 return
4047                 case "-b": builtin = true
4048                 case "-d": debug = true
4049                 case "-s": silent = true
4050                 case "-v": info = true
4051             }
4052             opt = arg1
4053             argv = argv[1:]
4054         }
4055     }
4056     dicName := ""
4057     dicURL := ""
4058     if 0 < len(argv) {
4059         arg = argv[0]
4060         dicName = arg
4061         argv = argv[1:]
4062     }
4063     if 0 < len(argv) {
4064         dicURL = argv[0]
4065         argv = argv[1:]
4066     }
4067     if false {
4068         fprintf(stderr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
4069     }
4070     if cmd == DIC_COM_LOAD {
4071         //dicType := ""
4072         dicBody := ""
4073         if !builtin && dicName != "" && dicURL == "" {
4074             f,err := os.Open(dicName)
4075             if err == nil {
4076                 dicURL = dicName
4077             }else{
4078                 f,err = os.Open(dicName+".html")
4079                 if err == nil {
4080                     dicURL = dicName+".html"
4081                 }else{
4082                     f,err = os.Open("gshdic-"+dicName+".html")
4083                     if err == nil {
4084                         dicURL = "gshdic-"+dicName+".html"
4085                     }
4086                 }
4087             }
4088             if err == nil {
4089                 var buf = make([]byte,128*1024)
4090                 count,err := f.Read(buf)
4091                 f.Close()
4092             }

```

```

4093         if info {
4094             fprintf(stderr, "--Id-- ReadDic(%v,%v)\n",count,err)
4095         }
4096         dicBody = string(buf[0:count])
4097     }
4098 }
4099 if dicBody == "" {
4100     switch arg {
4101     default:
4102         dicName = "WorldDic"
4103         dicURL = WorldDic
4104         if info {
4105             fprintf(stderr, "--Id-- default dictionary \"%v\"\n",
4106                 dicName);
4107         }
4108     case "wnn":
4109         dicName = "WnnDic"
4110         dicURL = WnnDic
4111     case "sumomo":
4112         dicName = "SumomoDic"
4113         dicURL = SumomoDic
4114     case "sijimi":
4115         dicName = "SijimiDic"
4116         dicURL = SijimiDic
4117     case "jkl":
4118         dicName = "JKLJaDic"
4119         dicURL = JA_JKLDic
4120     }
4121     if debug {
4122         fprintf(stderr, "--Id-- %v URL=%v\n\n",dicName,dicURL);
4123     }
4124     dicv := strings.Split(dicURL,",")
4125     if debug {
4126         fprintf(stderr, "--Id-- %v encoded data...\n",dicName)
4127         fprintf(stderr, "Type: %v\n",dicv[0])
4128         fprintf(stderr, "Body: %v\n",dicv[1])
4129         fprintf(stderr, "\n")
4130     }
4131     body,_ := base64.StdEncoding.DecodeString(dicv[1])
4132     dicBody = string(body)
4133 }
4134 if info {
4135     fmt.Printf("--Id-- %v %v\n",dicName,dicURL)
4136     fmt.Printf("%s\n",dicBody)
4137 }
4138 if debug {
4139     fprintf(stderr, "--Id-- dicName %v text...\n",dicName)
4140     fprintf(stderr, "%v\n",string(dicBody))
4141 }
4142 entv := strings.Split(dicBody, "\n");
4143 if info {
4144     fprintf(stderr, "--Id-- %v scan...\n",dicName);
4145 }
4146 var added int = 0
4147 var dup int = 0
4148 for i,v := range entv {
4149     var pat string
4150     var out string
4151     fmt.Sscanf(v,"%s %s",&pat,&out)
4152     if len(pat) <= 0 {
4153     }else{
4154         if 0 <= isinDic(pat) {
4155             dup += 1
4156             continue
4157         }
4158         romkana[dicents] = RomKana{dicName,pat,out,0}
4159         dicents += 1
4160         added += 1
4161         Romkan = append(Romkan,RomKana{dicName,pat,out,0})
4162         if debug {
4163             fmt.Printf("[%3v]:[%2v]%-8v [%2v]%-8v\n",
4164                 i,len(pat),pat,len(out),out)
4165         }
4166     }
4167 }
4168 if !silent {
4169     url := dicURL
4170     if strBegins(url,"data:") {
4171         url = "builtin"
4172     }
4173     fprintf(stderr, "--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4174         dicName,added,dup,len(Romkan),url);
4175 }
4176 // should sort by pattern length for conplete match, for performance
4177 if debug {
4178     arg = "" // search pattern
4179     dump = true
4180 }
4181 }
4182 if cmd == DIC_COM_DUMP || dump {
4183     fprintf(stderr, "--Id-- %v dump... %v entries:\n",dicName,len(Romkan));
4184     var match = 0
4185     for i := 0; i < len(Romkan); i++ {
4186         dic := Romkan[i].dic
4187         pat := Romkan[i].pat
4188         out := Romkan[i].out
4189         if arg == "" || 0 <= strings.Index(pat,arg)|| 0 <= strings.Index(out,arg) {
4190             fmt.Printf("\\\\%v\\t%v [%2v]%-8v [%2v]%-8v\n",
4191                 i,dic,len(pat),pat,len(out),out)
4192             match += 1
4193         }
4194     }
4195     fprintf(stderr, "--Id-- %v matched %v / %v entries:\n",arg,match,len(Romkan));
4196 }
4197 }
4198 func loadDefaultDic(dic int){
4199     if( 0 < len(Romkan) ){
4200         return
4201     }
4202     //fprintf(stderr, "\r\n")
4203     xDic([]string{"dic",DIC_COM_LOAD});
4204 }
4205 var info = false
4206 if info {
4207     fprintf(stderr, "--Id-- Conguraturations!! WorldDic is now activated.\r\n")
4208     fprintf(stderr, "--Id-- enter \"dic\" command for help.\r\n")
4209 }
4210 }
4211 func readDic()(int){
4212     /*
4213     var rk *os.File;
4214     var dic = "MyIME-dic.txt";
4215     //rk = fopen("romkana.txt", "r");
4216     //rk = fopen("JK-JA-morse-dic.txt", "r");

```

```

4217 rk = fopen(dic, "r");
4218 if( rk == NULL_FP ){
4219     if( true ){
4220         fprintf(stderr, "--%s-- Could not load %s\n", MyIMEVER, dic);
4221     }
4222     return -1;
4223 }
4224 if( true ){
4225     var di int;
4226     var line = make(StrBuff, 1024);
4227     var pat string
4228     var out string
4229     for di = 0; di < 1024; di++ {
4230         if( fgets(line, sizeof(line), rk) == NULLSP ){
4231             break;
4232         }
4233         fmt.Sscanf(string(line[0:strlen(line)]), "%s %s", &pat, &out);
4234         //sscanf(line, "%s %s", &pat, &out);
4235         romkana[di].pat = pat;
4236         romkana[di].out = out;
4237         //fprintf(stderr, "--Dd- %-10s %s\n", pat, out)
4238     }
4239     dicents += di
4240     if( false ){
4241         fprintf(stderr, "--%s-- loaded romkana.txt [%d]\n", MyIMEVER, di);
4242         for di = 0; di < dicents; di++ {
4243             fprintf(stderr,
4244                 "%s %s\n", romkana[di].pat, romkana[di].out);
4245         }
4246     }
4247 }
4248 fclose(rk);
4249
4250 //romkana[dicents].pat = "//ddump"
4251 //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
4252 */
4253 return 0;
4254 }
4255 func matchlen(stri string, pati string)(int){
4256     if strBegins(stri, pati) {
4257         return len(pati)
4258     }else{
4259         return 0
4260     }
4261 }
4262 func convs(src string)(string){
4263     var si int;
4264     var sx = len(src);
4265     var di int;
4266     var mi int;
4267     var dstb []byte
4268
4269     for si = 0; si < sx; { // search max. match from the position
4270         if strBegins(src[si:], "%x/") {
4271             // %x/integer/ // s/a/b/
4272             ix := strings.Index(src[si+3:], "/")
4273             if 0 < ix {
4274                 var iv int = 0
4275                 //fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
4276                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4277                 sval := fmt.Sprintf("%x", iv)
4278                 bval := []byte(sval)
4279                 dstb = append(dstb, bval...)
4280                 si = si+3+ix+1
4281                 continue
4282             }
4283         }
4284         if strBegins(src[si:], "%d/") {
4285             // %d/integer/ // s/a/b/
4286             ix := strings.Index(src[si+3:], "/")
4287             if 0 < ix {
4288                 var iv int = 0
4289                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4290                 sval := fmt.Sprintf("%d", iv)
4291                 bval := []byte(sval)
4292                 dstb = append(dstb, bval...)
4293                 si = si+3+ix+1
4294                 continue
4295             }
4296         }
4297         if strBegins(src[si:], "%t") {
4298             now := time.Now()
4299             if true {
4300                 date := now.Format(time.Stamp)
4301                 dstb = append(dstb, []byte(date)...)
4302                 si = si+3
4303             }
4304             continue
4305         }
4306     }
4307     var maxlen int = 0;
4308     var len int;
4309     mi = -1;
4310     for di = 0; di < dicents; di++ {
4311         len = matchlen(src[si:], romkana[di].pat);
4312         if( maxlen < len ){
4313             maxlen = len;
4314             mi = di;
4315         }
4316     }
4317     if( 0 < maxlen ){
4318         out := romkana[mi].out;
4319         dstb = append(dstb, []byte(out)...);
4320         si += maxlen;
4321     }else{
4322         dstb = append(dstb, src[si])
4323         si += 1;
4324     }
4325     return string(dstb)
4326 }
4327 func trans(src string)(int){
4328     dst := convs(src);
4329     xputss(dst, stderr);
4330     return 0;
4331 }
4332
4333 //----- LINEEDIT
4334 // "?" at the top of the line means searching history
4335
4336 // should be compatilbe with Telnet
4337 const (
4338     EV_MODE     = 255
4339     EV_IDLE    = 254
4340     EV_TIMEOUT = 253

```

```

4341
4342     GO_UP      = 252 // k
4343     GO_DOWN   = 251 // j
4344     GO_RIGHT  = 250 // l
4345     GO_LEFT   = 249 // h
4346     DEL_RIGHT = 248 // x
4347     GO_TOPL   = 'A'-0x40 // 0
4348     GO_ENDL   = 'E'-0x40 // $
4349
4350     GO_TOPW   = 239 // b
4351     GO_ENDW   = 238 // e
4352     GO_NEXTW  = 237 // w
4353
4354     GO_FORWCH = 229 // f
4355     GO_PAIRCH = 228 // %
4356
4357     GO_DEL    = 219 // d
4358
4359     HI_SRCH_FW = 209 // /
4360     HI_SRCH_BK = 208 // ?
4361     HI_SRCH_RFW = 207 // n
4362     HI_SRCH_RBK = 206 // N
4363 )
4364
4365 // should return number of octets ready to be read immediately
4366 //fprintf(stderr, "\n--Select(%v %v)\n", err, r.Bits[0])
4367
4368
4369 var EventRecvFd = -1 // file descriptor
4370 var EventSendFd = -1
4371 const EventFdOffset = 1000000
4372 const NormalFdOffset = 100
4373
4374 func putEvent(event int, evarg int){
4375     if true {
4376         if EventRecvFd < 0 {
4377             var pv = []int{-1,-1}
4378             syscall.Pipe(pv)
4379             EventRecvFd = pv[0]
4380             EventSendFd = pv[1]
4381             //fmt.Printf("--De-- EventPipe created[%v,%v]\n", EventRecvFd, EventSendFd)
4382         }
4383     }else{
4384         if EventRecvFd < 0 {
4385             // the document differs from this spec
4386             // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
4387             sv, err := syscall.Socketpair(syscall.AF_UNIX, syscall.SOCK_STREAM, 0)
4388             EventRecvFd = sv[0]
4389             EventSendFd = sv[1]
4390             if err != nil {
4391                 fmt.Printf("--De-- EventSock created[%v,%v] (%v)\n",
4392                     EventRecvFd, EventSendFd, err)
4393             }
4394         }
4395     }
4396     var buf = []byte{ byte(event)}
4397     n, err := syscall.Write(EventSendFd, buf)
4398     if err != nil {
4399         fmt.Printf("--De-- putEvent[%v] (%3v) (%v %v)\n", EventSendFd, event, n, err)
4400     }
4401 }
4402 func ungets(str string){
4403     for _, ch := range str {
4404         putEvent(int(ch), 0)
4405     }
4406 }
4407 func (gsh*GshContext)xReplay(argv []string){
4408     hix := 0
4409     tempo := 1.0
4410     xtempo := 1.0
4411     repeat := 1
4412
4413     for _, a := range argv { // tempo
4414         if strBegins(a, "x") {
4415             fmt.Sscanf(a[1:], "%f", &xtempo)
4416             tempo = 1 / xtempo
4417             //fprintf(stderr, "--Dr-- tempo=[%v] %v\n", a[2:], tempo);
4418         }else
4419         if strBegins(a, "r") { // repeat
4420             fmt.Sscanf(a[1:], "%v", &repeat)
4421         }else
4422         if strBegins(a, "!") {
4423             fmt.Sscanf(a[1:], "%d", &hix)
4424         }else{
4425             fmt.Sscanf(a, "%d", &hix)
4426         }
4427     }
4428     if hix == 0 || len(argv) <= 1 {
4429         hix = len(gsh.CommandHistory)-1
4430     }
4431     fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n", hix, xtempo, repeat)
4432     //dumpEvents(hix)
4433     //gsh.xScanReplay(hix, false, repeat, tempo, argv)
4434     go gsh.xScanReplay(hix, true, repeat, tempo, argv)
4435 }
4436
4437 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4438 // 2020-0827 GShell-0.2.3
4439 /*
4440 func FpollIn1(fp *os.File, usec int)(uintptr){
4441     nfd := 1
4442
4443     rdv := syscall.FdSet {}
4444     fd1 := fp.Fd()
4445     bank1 := fd1/32
4446     mask1 := int32(1 << fd1)
4447     rdv.Bits[bank1] = mask1
4448
4449     fd2 := -1
4450     bank2 := -1
4451     var mask2 int32 = 0
4452
4453     if 0 <= EventRecvFd {
4454         fd2 = EventRecvFd
4455         nfd = fd2 + 1
4456         bank2 = fd2/32
4457         mask2 = int32(1 << fd2)
4458         rdv.Bits[bank2] |= mask2
4459         //fmt.Printf("--De-- EventPoll mask added [%d][%v][%v]\n", fd2, bank2, mask2)
4460     }
4461
4462     tout := syscall.NsecToTimeval(int64(usec*1000))
4463     //n, err := syscall.Select(nfd, &rdv, nil, nil, &stout) // spec. mismatch
4464     err := syscall.Select(nfd, &rdv, nil, nil, &stout)

```

```

4465 if err != nil {
4466     //fmt.Printf("--De-- select() err(%v)\n",err)
4467 }
4468 if err == nil {
4469     if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4470         if false {
4471             fmt.Printf("--De-- got Event\n")
4472         }
4473         return uintptr(EventFdOffset + fd2)
4474     }else
4475     if (rdv.Bits[bank1] & mask1) != 0 {
4476         return uintptr(NormalFdOffset + fd1)
4477     }else{
4478         return 1
4479     }
4480 }else{
4481     return 0
4482 }
4483 }
4484 */
4485 func fgetcTimeoutl(fp *os.File,usec int)(int){
4486     READl:
4487     //readyFd := FpollInl(fp,usec)
4488     readyFd := CPpollInl(fp,usec)
4489     if readyFd < 100 {
4490         return EV_TIMEOUT
4491     }
4492 }
4493 var buf [1]byte
4494
4495 if EventFdOffset <= readyFd {
4496     fd := int(readyFd-EventFdOffset)
4497     _,err := syscall.Read(fd,buf[0:1])
4498     if( err != nil ){
4499         return EOF;
4500     }else{
4501         if buf[0] == EV_MODE {
4502             recvEvent(fd)
4503             goto READl
4504         }
4505         return int(buf[0])
4506     }
4507 }
4508
4509 _,err := fp.Read(buf[0:1])
4510 if( err != nil ){
4511     return EOF;
4512 }else{
4513     return int(buf[0])
4514 }
4515 }
4516
4517 func visibleChar(ch int)(string){
4518     switch {
4519     case '!' <= ch && ch <= '-':
4520         return string(ch)
4521     }
4522     switch ch {
4523     case '\': return "\\s"
4524     case '\n': return "\\n"
4525     case '\r': return "\\r"
4526     case '\t': return "\\t"
4527     }
4528     switch ch {
4529     case 0x00: return "NUL"
4530     case 0x07: return "BEL"
4531     case 0x08: return "BS"
4532     case 0x0E: return "SO"
4533     case 0x0F: return "SI"
4534     case 0x1B: return "ESC"
4535     case 0x7F: return "DEL"
4536     }
4537     switch ch {
4538     case EV_IDLE: return fmt.Sprintf("IDLE")
4539     case EV_MODE: return fmt.Sprintf("MODE")
4540     }
4541     return fmt.Sprintf("%X",ch)
4542 }
4543 func recvEvent(fd int){
4544     var buf = make([]byte,1)
4545     _,_ = syscall.Read(fd,buf[0:1])
4546     if( buf[0] != 0 ){
4547         romkanmode = true
4548     }else{
4549         romkanmode = false
4550     }
4551 }
4552 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
4553     var Start time.Time
4554     var events = []Event{}
4555     for _,e := range Events {
4556         if hix == 0 || e.CmdIndex == hix {
4557             events = append(events,e)
4558         }
4559     }
4560     elen := len(events)
4561     if 0 < elen {
4562         if events[elen-1].event == EV_IDLE {
4563             events = events[0:elen-1]
4564         }
4565     }
4566     for r := 0; r < repeat; r++ {
4567         for i,e := range events {
4568             nano := e.when.Nanosecond()
4569             micro := nano / 1000
4570             if Start.Second() == 0 {
4571                 Start = time.Now()
4572             }
4573             diff := time.Now().Sub(Start)
4574             if replay {
4575                 if e.event != EV_IDLE {
4576                     putEvent(e.event,0)
4577                     if e.event == EV_MODE { // event with arg
4578                         putEvent(int(e.evarg),0)
4579                     }
4580                 }
4581             }else{
4582                 fmt.Printf("%7.3fms %#-3v !%-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",
4583                     float64(diff)/1000000.0,
4584                     i,
4585                     e.CmdIndex,
4586                     e.when.Format(time.Stamp),micro,
4587                     e.event,e.event,visibleChar(e.event),
4588                     float64(e.evarg)/1000000.0)

```

```

4589     }
4590     if e.event == EV_IDLE {
4591         d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
4592         //nsleep(time.Duration(e.evarg))
4593         nsleep(d)
4594     }
4595     }
4596 }
4597 }
4598 func dumpEvents(arg[]string){
4599     hix := 0
4600     if 1 < len(arg) {
4601         fmt.Sprintf(arg[1], "%d", hix)
4602     }
4603     for i, e := range Events {
4604         nano := e.when.Nanosecond()
4605         micro := nano / 1000
4606         //if e.event != EV_TIMEOUT {
4607         if hix == 0 || e.CmdIndex == hix {
4608             fmt.Printf("#%-3v !%-3v [%v.%06d] %3v %02X %-4v %10.3fms\n", i,
4609                 e.CmdIndex,
4610                 e.when.Format(time.Stamp), micro,
4611                 e.event, e.event, visibleChar(e.event), float64(e.evarg)/1000000.0)
4612         }
4613         //}
4614     }
4615 }
4616 func fgetcTimeout(fp *os.File, usec int)(int){
4617     ch := fgetcTimeout1(fp, usec)
4618     if ch != EV_TIMEOUT {
4619         now := time.Now()
4620         if 0 < len(Events) {
4621             last := Events[len(Events)-1]
4622             dura := int64(now.Sub(last.when))
4623             Events = append(Events, Event{last.when, EV_IDLE, dura, last.CmdIndex})
4624         }
4625         Events = append(Events, Event{time.Now(), ch, 0, CmdIndex})
4626     }
4627     return ch
4628 }
4629 }
4630 var AtConsoleLineTop = true
4631 var TtyMaxCol = 72 // to be obtained by ioctl?
4632 var EscTimeout = (100*1000)
4633 var (
4634     MODE_VicMode    bool // vi compatible command mode
4635     MODE_ShowMode  bool
4636     romkanmode     bool // shown translation mode, the mode to be retained
4637     MODE_Recursive bool // recursive translation
4638     MODE_CapsLock  bool // software CapsLock
4639     MODE_LowerLock bool // force lower-case character lock
4640     MODE_ViInsert  int // visible insert mode, should be like "I" icon in X Window
4641     MODE_ViTrace   bool // output newline before translation
4642 )
4643 type IInput struct {
4644     lno int
4645     lastlno int
4646     pch []int // input queue
4647     prompt string
4648     line string
4649     right string
4650     inJmode bool
4651     pinJmode bool
4652     waitingMeta string // waiting meta character
4653     LastCmd string
4654 }
4655 func (iin*IInput)Getc(timeoutUs int)(int){
4656     ch1 := EOF
4657     ch2 := EOF
4658     ch3 := EOF
4659     if( 0 < len(iin.pch) ){ // deQ
4660         ch1 = iin.pch[0]
4661         iin.pch = iin.pch[1:]
4662     }else{
4663         ch1 = fgetcTimeout(stdin, timeoutUs);
4664     }
4665     if( ch1 == 033 ){ // escape sequence
4666         ch2 = fgetcTimeout(stdin, EscTimeout);
4667         if( ch2 == EV_TIMEOUT ){
4668         }else{
4669             ch3 = fgetcTimeout(stdin, EscTimeout);
4670             if( ch3 == EV_TIMEOUT ){
4671                 iin.pch = append(iin.pch, ch2) // enQ
4672             }else{
4673                 switch( ch2 ){
4674                     default:
4675                         iin.pch = append(iin.pch, ch2) // enQ
4676                         iin.pch = append(iin.pch, ch3) // enQ
4677                     case ' ':
4678                         switch( ch3 ){
4679                             case 'A': ch1 = GO_UP; // ^
4680                             case 'B': ch1 = GO_DOWN; // v
4681                             case 'C': ch1 = GO_RIGHT; // >
4682                             case 'D': ch1 = GO_LEFT; // <
4683                             case '3':
4684                                 ch4 := fgetcTimeout(stdin, EscTimeout);
4685                                 if( ch4 == '-' ){
4686                                     //fprintf(stderr, "x[%02X %02X %02X %02X]\n", ch1, ch2, ch3, ch4);
4687                                     ch1 = DEL_RIGHT
4688                                 }
4689                             }
4690                     case '\\':
4691                         //ch4 := fgetcTimeout(stdin, EscTimeout);
4692                         //fprintf(stderr, "y[%02X %02X %02X %02X]\n", ch1, ch2, ch3, ch4);
4693                         switch( ch3 ){
4694                             case '-': ch1 = DEL_RIGHT
4695                         }
4696                     }
4697             }
4698         }
4699     }
4700     return ch1
4701 }
4702 func (iin*IInput)clearline(){
4703     var i int
4704     fprintf(stderr, "\r");
4705     // should be ANSI ESC sequence
4706     for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
4707         fputc(' ', os.Stderr);
4708     }
4709     fprintf(stderr, "\r");
4710 }
4711 func (iin*IInput)Redraw(){
4712     redraw(iin, iin.lno, iin.line, iin.right)

```

```

4713 }
4714 func redraw(iin *IInput,lno int,line string,right string){
4715     inMeta := false
4716     showMode := ""
4717     showMeta := "" // visible Meta mode on the cursor position
4718     showLino := fmt.Sprintf("%d!",lno)
4719     InsertMark := "" // in visible insert mode
4720
4721     if MODE_VicMode {
4722     }else
4723     if 0 < len(iin.right) {
4724         InsertMark = " "
4725     }
4726
4727     if( 0 < len(iin.waitingMeta) ){
4728         inMeta = true
4729         if iin.waitingMeta[0] != 033 {
4730             showMeta = iin.waitingMeta
4731         }
4732     }
4733     if( romkanmode ){
4734         //romkanmark = " *";
4735     }else{
4736         //romkanmark = "";
4737     }
4738     if MODE_ShowMode {
4739         romkan := "--"
4740         inmeta := "-"
4741         inveri := ""
4742         if MODE_CapsLock {
4743             inmeta = "A"
4744         }
4745         if MODE_LowerLock {
4746             inmeta = "a"
4747         }
4748         if MODE_ViTrace {
4749             inveri = "v"
4750         }
4751         if MODE_VicMode {
4752             inveri = ":"
4753         }
4754         if romkanmode {
4755             romkan = "\343\201\202"
4756             if MODE_CapsLock {
4757                 inmeta = "R"
4758             }else{
4759                 inmeta = "r"
4760             }
4761         }
4762         if inMeta {
4763             inmeta = "\\ "
4764         }
4765         showMode = "["+romkan+inmeta+inveri+"]";
4766     }
4767     Pre := "\r" + showMode + showLino
4768     Output := ""
4769     Left := ""
4770     Right := ""
4771     if romkanmode {
4772         Left = convs(line)
4773         Right = InsertMark+convs(right)
4774     }else{
4775         Left = line
4776         Right = InsertMark+right
4777     }
4778     Output = Pre+Left
4779     if MODE_ViTrace {
4780         Output += iin.LastCmd
4781     }
4782     Output += showMeta+Right
4783     for len(Output) < TtyMaxCol { // to the max. position that may be dirty
4784         Output += " "
4785         // should be ANSI ESC sequence
4786         // not necessary just after newline
4787     }
4788     Output += Pre+Left+showMeta // to set the cursor to the current input position
4789     fprintf(stderr,"%s",Output)
4790
4791     if MODE_ViTrace {
4792         if 0 < len(iin.LastCmd) {
4793             iin.LastCmd = ""
4794             fprintf(stderr,"\r\n")
4795         }
4796     }
4797     AtConsoleLineTop = false
4798 }
4799 // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
4800 func delHeadChar(str string)(rline string,head string){
4801     _,clen := utf8.DecodeRune([]byte(str))
4802     head = string(str[0:clen])
4803     return str[clen:],head
4804 }
4805 func delTailChar(str string)(rline string, last string){
4806     var i = 0
4807     var clen = 0
4808     for {
4809         _,siz := utf8.DecodeRune([]byte(str)[i:])
4810         if siz <= 0 { break }
4811         clen = siz
4812         i += siz
4813     }
4814     last = str[len(str)-clen:]
4815     return str[0:len(str)-clen],last
4816 }
4817 // 3> for output and history
4818 // 4> for keylog?
4819 // <a name="getline">Command Line Editor</a>
4820 func xgetline(lno int, prevline string, gsh*GshContext)(string){
4821     var iin IInput
4822     iin.lastlno = lno
4823     iin.lno = lno
4824
4825     CmdIndex = len(gsh.CommandHistory)
4826     if( isatty(0) == 0 ){
4827         if( sfgets(&iin.line,LINESIZE,stdin) == NULL ){
4828             iin.line = "exit\n";
4829         }else{
4830         }
4831     }
4832     return iin.line
4833 }
4834 if( true ){
4835     //var pts string;
4836     //pts = ptsname(0);

```

```

4837 //pts = ttyname(0);
4838 //fprintf(stderr,"--pts[0] = %s\n",pts?pts:"?");
4839 }
4840 if( false ){
4841     fprintf(stderr,"! ");
4842     fflush(stderr);
4843     sfgets(&iin.line,LINESIZE,stdin);
4844     return iin.line
4845 }
4846 system("/bin/stty -echo -icanon");
4847 xline := iin.xgetline(prevline,gsh)
4848 system("/bin/stty echo sane");
4849 return xline
4850 }
4851 func (iin*IInput)Translate(cmdch int){
4852     romkanmode = !romkanmode;
4853     if MODE_ViTrace {
4854         fprintf(stderr,"%v\r\n",string(cmdch));
4855     }else
4856     if( cmdch == 'J' ){
4857         fprintf(stderr,"J\r\n");
4858         iin.inJmode = true
4859     }
4860     iin.Redraw();
4861     loadDefaultDic(cmdch);
4862     iin.Redraw();
4863 }
4864 func (iin*IInput)Replace(cmdch int){
4865     iin.LastCmd = fmt.Sprintf("%v",string(cmdch))
4866     iin.Redraw();
4867     loadDefaultDic(cmdch);
4868     dst := convs(iin.line+iin.right);
4869     iin.line = dst
4870     iin.right = ""
4871     if( cmdch == 'I' ){
4872         fprintf(stderr,"I\r\n");
4873         iin.inJmode = true
4874     }
4875     iin.Redraw();
4876 }
4877 // aa 12 alal
4878 func isAlpha(ch rune)(bool){
4879     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
4880         return true
4881     }
4882     return false
4883 }
4884 func isAlnum(ch rune)(bool){
4885     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
4886         return true
4887     }
4888     if '0' <= ch && ch <= '9' {
4889         return true
4890     }
4891     return false
4892 }
4893 }
4894 // 0.2.8 2020-0901 created
4895 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
4896 func (iin*IInput)GotoTOPW(){
4897     str := iin.line
4898     i := len(str)
4899     if i <= 0 {
4900         return
4901     }
4902     //i0 := i
4903     i -= 1
4904     lastSize := 0
4905     var lastRune rune
4906     var found = -1
4907     for 0 < i { // skip preamble spaces
4908         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4909         if !isAlnum(lastRune) { // character, type, or string to be searched
4910             i -= lastSize
4911             continue
4912         }
4913         break
4914     }
4915     for 0 < i {
4916         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4917         if lastSize <= 0 { continue } // not the character top
4918         if !isAlnum(lastRune) { // character, type, or string to be searched
4919             found = i
4920             break
4921         }
4922         i -= lastSize
4923     }
4924     if found < 0 && i == 0 {
4925         found = 0
4926     }
4927     if 0 <= found {
4928         if isAlnum(lastRune) { // or non-kana character
4929             }else{ // when positioning to the top o the word
4930                 i += lastSize
4931             }
4932         iin.right = str[i:] + iin.right
4933         if 0 < i {
4934             iin.line = str[0:i]
4935         }else{
4936             iin.line = ""
4937         }
4938     }
4939     //fmt.Printf("\n(%d,%d,%d)[%s][%s]\n",i0,i,found,iin.line,iin.right)
4940     //fmt.Printf("") // set debug messae at the end of line
4941 }
4942 // 0.2.8 2020-0901 created
4943 func (iin*IInput)GotoENDW(){
4944     str := iin.right
4945     if len(str) <= 0 {
4946         return
4947     }
4948     lastSize := 0
4949     var lastRune rune
4950     var lastW = 0
4951     i := 0
4952     inWord := false
4953
4954     lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
4955     if isAlnum(lastRune) {
4956         r,z := utf8.DecodeRuneInString(str[lastSize:])
4957         if 0 < z && isAlnum(r) {
4958             inWord = true
4959         }
4960     }

```



```

4961 for i < len(str) {
4962     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4963     if lastSize <= 0 { break } // broken data?
4964     if !isAlnum(lastRune) { // character, type, or string to be searched
4965         break
4966     }
4967     lastW = i // the last alnum if in alnum word
4968     i += lastSize
4969 }
4970 if inWord {
4971     goto DISP
4972 }
4973 for i < len(str) {
4974     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4975     if lastSize <= 0 { break } // broken data?
4976     if isAlnum(lastRune) { // character, type, or string to be searched
4977         break
4978     }
4979     i += lastSize
4980 }
4981 for i < len(str) {
4982     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
4983     if lastSize <= 0 { break } // broken data?
4984     if !isAlnum(lastRune) { // character, type, or string to be searched
4985         break
4986     }
4987     lastW = i
4988     i += lastSize
4989 }
4990 DISP:
4991 if 0 < lastW {
4992     iin.line = iin.line + str[0:lastW]
4993     iin.right = str[lastW:]
4994 }
4995 //fmt.Printf("\n%d[%s][%s]\n",i,iin.line,iin.right)
4996 //fmt.Printf("") // set debug messae at the end of line
4997 }
4998 // 0.2.8 2020-0901 created
4999 func (iin*Input)GotoNEXTW(){
5000     str := iin.right
5001     if len(str) <= 0 {
5002         return
5003     }
5004     lastSize := 0
5005     var lastRune rune
5006     var found = -1
5007     i := 1
5008     for i < len(str) {
5009         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5010         if lastSize <= 0 { break } // broken data?
5011         if !isAlnum(lastRune) { // character, type, or string to be searched
5012             found = i
5013             break
5014         }
5015         i += lastSize
5016     }
5017     if 0 < found {
5018         if isAlnum(lastRune) { // or non-kana character
5019             }else{ // when positioning to the top o the word
5020                 found += lastSize
5021             }
5022             iin.line = iin.line + str[0:found]
5023             if 0 < found {
5024                 iin.right = str[found:]
5025             }else{
5026                 iin.right = ""
5027             }
5028         }
5029         //fmt.Printf("\n%d[%s][%s]\n",i,iin.line,iin.right)
5030         //fmt.Printf("") // set debug messae at the end of line
5031     }
5032     // 0.2.8 2020-0902 created
5033     func (iin*Input)GotoPAIRCH(){
5034         str := iin.right
5035         if len(str) <= 0 {
5036             return
5037         }
5038         lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
5039         if lastSize <= 0 {
5040             return
5041         }
5042         forw := false
5043         back := false
5044         pair := ""
5045         switch string(lastRune){
5046             case "{": pair = "}"; forw = true
5047             case "}": pair = "{"; back = true
5048             case "(": pair = ")"; forw = true
5049             case ")": pair = "("; back = true
5050             case "[": pair = "]"; forw = true
5051             case "]": pair = "["; back = true
5052             case "<": pair = ">"; forw = true
5053             case ">": pair = "<"; back = true
5054             case "\'": pair = "\""; // context depednet, can be f' or back-double quote
5055             case '\"': pair = "'"; // context depednet, can be f' or back-quote
5056             // case Japanese Kakkos
5057         }
5058         if forw {
5059             iin.SearchForward(pair)
5060         }
5061         if back {
5062             iin.SearchBackward(pair)
5063         }
5064     }
5065     // 0.2.8 2020-0902 created
5066     func (iin*Input)SearchForward(pat string)(bool){
5067         right := iin.right
5068         found := -1
5069         i := 0
5070         if strBegins(right,pat) {
5071             _,z := utf8.DecodeRuneInString(right[i:])
5072             if 0 < z {
5073                 i += z
5074             }
5075         }
5076         for i < len(right) {
5077             if strBegins(right[i:],pat) {
5078                 found = i
5079                 break
5080             }
5081             _,z := utf8.DecodeRuneInString(right[i:])
5082             if z <= 0 { break }
5083             i += z
5084         }

```

```

5085     if 0 <= found {
5086         iin.line = iin.line + right[0:found]
5087         iin.right = iin.right[found:]
5088         return true
5089     }else{
5090         return false
5091     }
5092 }
5093 // 0.2.8 2020-0902 created
5094 func (iin*IInput)SearchBackward(pat string)(bool){
5095     line := iin.line
5096     found := -1
5097     i := len(line)-1
5098     for i = i; 0 <= i; i-- {
5099         _,z := utf8.DecodeRuneInString(line[i:])
5100         if z <= 0 {
5101             continue
5102         }
5103         //fprintf(stderr,"-- %v %v\n",pat,line[i:])
5104         if strBegins(line[i:],pat) {
5105             found = i
5106             break
5107         }
5108     }
5109     //fprintf(stderr,"--%d\n",found)
5110     if 0 <= found {
5111         iin.right = line[found:] + iin.right
5112         iin.line = line[0:found]
5113         return true
5114     }else{
5115         return false
5116     }
5117 }
5118 // 0.2.8 2020-0902 created
5119 // search from top, end, or current position
5120 func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool,string){
5121     if forw {
5122         for _,v := range gsh.CommandHistory {
5123             if 0 <= strings.Index(v.CmdLine,pat) {
5124                 //fprintf(stderr,"\n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
5125                 return true,v.CmdLine
5126             }
5127         }
5128     }else{
5129         hlen := len(gsh.CommandHistory)
5130         for i := hlen-1; 0 < i; i-- {
5131             v := gsh.CommandHistory[i]
5132             if 0 <= strings.Index(v.CmdLine,pat) {
5133                 //fprintf(stderr,"\n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
5134                 return true,v.CmdLine
5135             }
5136         }
5137     }
5138     //fprintf(stderr,"\n--De-- not-found(%v)\n",pat)
5139     return false,"(Not Found in History)"
5140 }
5141 // 0.2.8 2020-0902 created
5142 func (iin*IInput)GotoFORWSTR(pat string,gsh*GshContext){
5143     found := false
5144     if 0 < len(iin.right) {
5145         found = iin.SearchForward(pat)
5146     }
5147     if !found {
5148         found,line := gsh.SearchHistory(pat,true)
5149         if found {
5150             iin.line = line
5151             iin.right = ""
5152         }
5153     }
5154 }
5155 func (iin*IInput)GotoBACKSTR(pat string, gsh*GshContext){
5156     found := false
5157     if 0 < len(iin.line) {
5158         found = iin.SearchBackward(pat)
5159     }
5160     if !found {
5161         found,line := gsh.SearchHistory(pat,false)
5162         if found {
5163             iin.line = line
5164             iin.right = ""
5165         }
5166     }
5167 }
5168 func (iin*IInput)getString1(prompt string)(string){ // should be editable
5169     iin.clearline();
5170     fprintf(stderr,"\r%v",prompt)
5171     str := ""
5172     for {
5173         ch := iin.Getc(10*1000*1000)
5174         if ch == '\n' || ch == '\r' {
5175             break
5176         }
5177         sch := string(ch)
5178         str += sch
5179         fprintf(stderr,"%s",sch)
5180     }
5181     return str
5182 }
5183
5184 // search pattern must be an array and selectable with ^N/^P
5185 var SearchPat = ""
5186 var SearchForw = true
5187
5188 func (iin*IInput)xgetline1(prevline string, gsh*GshContext)(string){
5189     var ch int;
5190
5191     MODE_ShowMode = false
5192     MODE_VicMode = false
5193     iin.Redraw();
5194     first := true
5195
5196     for cix := 0; ; cix++ {
5197         iin.pinJmode = iin.inJmode
5198         iin.inJmode = false
5199
5200         ch = iin.Getc(1000*1000)
5201
5202         if ch != EV_TIMEOUT && first {
5203             first = false
5204             mode := 0
5205             if romkanmode {
5206                 mode = 1
5207             }
5208             now := time.Now()

```

```

5209     Events = append(Events, Event{now, EV_MODE, int64(mode), CmdIndex})
5210 }
5211 if ch == 033 {
5212     MODE_ShowMode = true
5213     MODE_VicMode = !MODE_VicMode
5214     iin.Redraw();
5215     continue
5216 }
5217 if MODE_VicMode {
5218     switch ch {
5219     case '0': ch = GO_TOPL
5220     case '$': ch = GO_ENDL
5221     case 'b': ch = GO_TOPW
5222     case 'e': ch = GO_ENDW
5223     case 'w': ch = GO_NEXTW
5224     case '%': ch = GO_PAIRCH
5225
5226     case 'j': ch = GO_DOWN
5227     case 'k': ch = GO_UP
5228     case 'h': ch = GO_LEFT
5229     case 'l': ch = GO_RIGHT
5230     case 'x': ch = DEL_RIGHT
5231     case 'a': MODE_VicMode = !MODE_VicMode
5232     case 'i': MODE_VicMode = !MODE_VicMode
5233     case ' ': ch = GO_RIGHT
5234     case 'i': MODE_VicMode = !MODE_VicMode
5235     case '-':
5236         right, head := delHeadChar(iin.right)
5237         if len([]byte(head)) == 1 {
5238             ch = int(head[0])
5239             if( 'a' <= ch && ch <= 'z' ){
5240                 ch = ch + 'A'-'a'
5241             }else
5242             if( 'A' <= ch && ch <= 'Z' ){
5243                 ch = ch + 'a'-'A'
5244             }
5245             iin.right = string(ch) + right
5246         }
5247         iin.Redraw();
5248         continue
5249     case 'f': // GO_FORWCH
5250     case 'f': // GO_FORWCH
5251         iin.Redraw();
5252         ch = iin.Getc(3*1000*1000)
5253         if ch == EV_TIMEOUT {
5254             iin.Redraw();
5255             continue
5256         }
5257         SearchPat = string(ch)
5258         SearchForw = true
5259         iin.GotoFORWSTR(SearchPat, gsh)
5260         iin.Redraw();
5261         continue
5262     case '/':
5263         SearchPat = iin.getstringl("/") // should be editable
5264         SearchForw = true
5265         iin.GotoFORWSTR(SearchPat, gsh)
5266         iin.Redraw();
5267         continue
5268     case '?':
5269         SearchPat = iin.getstringl("?") // should be editable
5270         SearchForw = false
5271         iin.GotoBACKSTR(SearchPat, gsh)
5272         iin.Redraw();
5273         continue
5274     case 'n':
5275         if SearchForw {
5276             iin.GotoFORWSTR(SearchPat, gsh)
5277         }else{
5278             iin.GotoBACKSTR(SearchPat, gsh)
5279         }
5280         iin.Redraw();
5281         continue
5282     case 'N':
5283         if !SearchForw {
5284             iin.GotoFORWSTR(SearchPat, gsh)
5285         }else{
5286             iin.GotoBACKSTR(SearchPat, gsh)
5287         }
5288         iin.Redraw();
5289         continue
5290     }
5291 }
5292 switch ch {
5293 case GO_TOPW:
5294     iin.GotoTOPW()
5295     iin.Redraw();
5296     continue
5297 case GO_ENDW:
5298     iin.GotoENDW()
5299     iin.Redraw();
5300     continue
5301 case GO_NEXTW:
5302     // to next space then
5303     iin.GotoNEXTW()
5304     iin.Redraw();
5305     continue
5306 case GO_PAIRCH:
5307     iin.GotoPAIRCH()
5308     iin.Redraw();
5309     continue
5310 }
5311 //fprintf(stderr, "A[%02X]\n", ch);
5312 if( ch == '\\\ | | ch == 033 ){
5313     MODE_ShowMode = true
5314     metach := ch
5315     iin.waitingMeta = string(ch)
5316     iin.Redraw();
5317     // set cursor //fprintf(stderr, "???\b\b\b")
5318     ch = fgetcTimeout(stdin, 2000*1000)
5319     // reset cursor
5320     iin.waitingMeta = ""
5321
5322     cmdch := ch
5323     if( ch == EV_TIMEOUT ){
5324         if metach == 033 {
5325             continue
5326         }
5327     }
5328     ch = metach
5329 }else
5330 /*
5331 if( ch == 'm' || ch == 'M' ){
5332     mch := fgetcTimeout(stdin, 1000*1000)

```

```

5333         if mch == 'r' {
5334             romkanmode = true
5335         }else{
5336             romkanmode = false
5337         }
5338         continue
5339     }else
5340     /*
5341     if( ch == 'k' || ch == 'K' ){
5342         MODE_Recursive = !MODE_Recursive
5343         iin.Translate(cmdch);
5344         continue
5345     }else
5346     if( ch == 'j' || ch == 'J' ){
5347         iin.Translate(cmdch);
5348         continue
5349     }else
5350     if( ch == 'i' || ch == 'I' ){
5351         iin.Replace(cmdch);
5352         continue
5353     }else
5354     if( ch == 'l' || ch == 'L' ){
5355         MODE_LowerLock = !MODE_LowerLock
5356         MODE_CapsLock = false
5357         if MODE_ViTrace {
5358             fprintf(stderr, "%v\r\n", string(cmdch));
5359         }
5360         iin.Redraw();
5361         continue
5362     }else
5363     if( ch == 'u' || ch == 'U' ){
5364         MODE_CapsLock = !MODE_CapsLock
5365         MODE_LowerLock = false
5366         if MODE_ViTrace {
5367             fprintf(stderr, "%v\r\n", string(cmdch));
5368         }
5369         iin.Redraw();
5370         continue
5371     }else
5372     if( ch == 'v' || ch == 'V' ){
5373         MODE_ViTrace = !MODE_ViTrace
5374         if MODE_ViTrace {
5375             fprintf(stderr, "%v\r\n", string(cmdch));
5376         }
5377         iin.Redraw();
5378         continue
5379     }else
5380     if( ch == 'c' || ch == 'C' ){
5381         if 0 < len(iin.line) {
5382             xline,tail := delTailChar(iin.line)
5383             if len([]byte(tail)) == 1 {
5384                 ch = int(tail[0])
5385                 if( 'a' <= ch && ch <= 'z' ){
5386                     ch = ch + 'A'-'a'
5387                 }else
5388                 if( 'A' <= ch && ch <= 'Z' ){
5389                     ch = ch + 'a'-'A'
5390                 }
5391                 iin.line = xline + string(ch)
5392             }
5393         }
5394         if MODE_ViTrace {
5395             fprintf(stderr, "%v\r\n", string(cmdch));
5396         }
5397         iin.Redraw();
5398         continue
5399     }else{
5400         iin.pch = append(iin.pch,ch) // push
5401         ch = '\\'
5402     }
5403 }
5404 switch( ch ){
5405     case 'P'-0x40: ch = GO_UP
5406     case 'N'-0x40: ch = GO_DOWN
5407     case 'B'-0x40: ch = GO_LEFT
5408     case 'F'-0x40: ch = GO_RIGHT
5409 }
5410 //fprintf(stderr, "B[%02X]\n",ch);
5411 switch( ch ){
5412     case 0:
5413         continue;
5414
5415     case '\t':
5416         iin.Replace('j');
5417         continue
5418     case 'X'-0x40:
5419         iin.Replace('j');
5420         continue
5421
5422     case EV_TIMEOUT:
5423         iin.Redraw();
5424         if iin.pinJmode {
5425             fprintf(stderr, "\\J\r\n")
5426             iin.inJmode = true
5427         }
5428         continue
5429     case GO_UP:
5430         if iin.lno == 1 {
5431             continue
5432         }
5433         cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
5434         if ok {
5435             iin.line = cmd
5436             iin.right = ""
5437             iin.lno = iin.lno - 1
5438         }
5439         iin.Redraw();
5440         continue
5441     case GO_DOWN:
5442         cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
5443         if ok {
5444             iin.line = cmd
5445             iin.right = ""
5446             iin.lno = iin.lno + 1
5447         }else{
5448             iin.line = ""
5449             iin.right = ""
5450             if iin.lno == iin.lastlno-1 {
5451                 iin.lno = iin.lno + 1
5452             }
5453         }
5454         iin.Redraw();
5455         continue
5456     case GO_LEFT:

```

```

5457         if 0 < len(iin.line) {
5458             xline,tail := delTailChar(iin.line)
5459             iin.line = xline
5460             iin.right = tail + iin.right
5461         }
5462         iin.Redraw();
5463         continue;
5464     case GO_RIGHT:
5465         if( 0 < len(iin.right) && iin.right[0] != 0 ){
5466             xright,head := delHeadChar(iin.right)
5467             iin.right = xright
5468             iin.line += head
5469         }
5470         iin.Redraw();
5471         continue;
5472     case EOF:
5473         goto EXIT; // replace
5474     case 'R'-0x40: // replace
5475         dst := convs(iin.line+iin.right);
5476         iin.line = dst
5477         iin.right = ""
5478         iin.Redraw();
5479         continue;
5480     case 'T'-0x40: // just show the result
5481         readDic();
5482         romkanmode = !romkanmode;
5483         iin.Redraw();
5484         continue;
5485     case 'L'-0x40:
5486         iin.Redraw();
5487         continue;
5488     case 'K'-0x40:
5489         iin.right = ""
5490         iin.Redraw();
5491         continue;
5492     case 'E'-0x40:
5493         iin.line += iin.right
5494         iin.right = ""
5495         iin.Redraw();
5496         continue;
5497     case 'A'-0x40:
5498         iin.right = iin.line + iin.right
5499         iin.line = ""
5500         iin.Redraw();
5501         continue;
5502     case 'U'-0x40:
5503         iin.line = ""
5504         iin.right = ""
5505         iin.clearline();
5506         iin.Redraw();
5507         continue;
5508     case DEL_RIGHT:
5509         if( 0 < len(iin.right) ){
5510             iin.right,_ = delHeadChar(iin.right)
5511             iin.Redraw();
5512         }
5513         continue;
5514     case 0x7F: // BS? not DEL
5515         if( 0 < len(iin.line) ){
5516             iin.line,_ = delTailChar(iin.line)
5517             iin.Redraw();
5518         }
5519         /*
5520         else
5521             if( 0 < len(iin.right) ){
5522                 iin.right,_ = delHeadChar(iin.right)
5523                 iin.Redraw();
5524             }
5525         */
5526         continue;
5527     case 'H'-0x40:
5528         if( 0 < len(iin.line) ){
5529             iin.line,_ = delTailChar(iin.line)
5530             iin.Redraw();
5531         }
5532         continue;
5533     }
5534     if( ch == '\n' || ch == '\r' ){
5535         iin.line += iin.right;
5536         iin.right = ""
5537         iin.Redraw();
5538         fputc(ch,stderr);
5539         AtConsoleLineTop = true
5540         break;
5541     }
5542     if MODE_CapsLock {
5543         if 'a' <= ch && ch <= 'z' {
5544             ch = ch+'A'-'a'
5545         }
5546     }
5547     if MODE_LowerLock {
5548         if 'A' <= ch && ch <= 'Z' {
5549             ch = ch+'a'-'A'
5550         }
5551     }
5552     iin.line += string(ch);
5553     iin.Redraw();
5554 }
5555 EXIT:
5556     return iin.line + iin.right;
5557 }
5558
5559 func getline_main(){
5560     line := xgetline(0,"",nil)
5561     fprintf(stderr,"%s\n",line);
5562 }
5563 /*
5564 dp = strpbrk(line,"\r\n");
5565 if( dp != NULL ){
5566     *dp = 0;
5567 }
5568 if( 0 ){
5569     fprintf(stderr,"\n(%d)\n",int(strlen(line)));
5570 }
5571 if( lseek(3,0,0) == 0 ){
5572     if( romkanmode ){
5573         var buf [8*1024]byte;
5574         convs(line,buf);
5575         strcpy(line,buf);
5576     }
5577     write(3,line,strlen(line));
5578     ftruncate(3,lseek(3,0,SEEK_CUR));
5579     //fprintf(stderr,"outsize=%d\n",int)lseek(3,0,SEEK_END));
5580     lseek(3,0,SEEK_SET);

```

```

5581     close(3);
5582 }else{
5583     fprintf(stderr, "\r\n gotline: ");
5584     trans(line);
5585     //printf("%s\n", line);
5586     printf("\n");
5587 }
5588 */
5589 }
5590 //== end ===== getline
5591
5592 //
5593 // $USERHOME/.gsh/
5594 // gsh-ro.txt, or gsh-configure.txt
5595 // gsh-history.txt
5596 // gsh-aliases.txt // should be conditional?
5597 //
5598 func (gshCtx *GshContext)gshSetupHomedir()(bool) {
5599     homedir, found := userHomeDir()
5600     if !found {
5601         fmt.Printf("--E-- You have no UserHomeDir\n")
5602         return true
5603     }
5604     gshhome := homedir + "/" + GSH_HOME
5605     _, err2 := os.Stat(gshhome)
5606     if err2 != nil {
5607         err3 := os.Mkdir(gshhome, 0700)
5608         if err3 != nil {
5609             fmt.Printf("--E-- Could not Create %s (%s)\n",
5610                 gshhome, err3)
5611             return true
5612         }
5613         fmt.Printf("--I-- Created %s\n", gshhome)
5614     }
5615     gshCtx.GshHomeDir = gshhome
5616     return false
5617 }
5618 func setupGshContext()(GshContext, bool){
5619     gshPA := syscall.ProcAttr {
5620         "", // the staring directory
5621         os.Environ(), // environ[]
5622         []uintptr{os.Stdin.Fd(), os.Stdout.Fd(), os.Stderr.Fd()},
5623         nil, // OS specific
5624     }
5625     cwd, _ := os.Getwd()
5626     gshCtx := GshContext {
5627         cwd, // StartDir
5628         "", // GetLine
5629         []GchdirHistory { {cwd, time.Now(), 0} }, // ChdirHistory
5630         gshPA,
5631         []GCommandHistory {}, // something for invokation?
5632         GCommandHistory {}, // CmdCurrent
5633         false,
5634         []int {},
5635         syscall.Rusage {},
5636         "", // GshHomeDir
5637         Ttyid {},
5638         false,
5639         false,
5640         []PluginInfo {},
5641         []string {},
5642         "",
5643         "v",
5644         ValueStack {},
5645         GServer {"", ""}, // LastServer
5646         "", // RSERVER
5647         cwd, // RWD
5648         CheckSum {},
5649     }
5650     err := gshCtx.gshSetupHomedir()
5651     return gshCtx, err
5652 }
5653 func (gsh *GshContext)gshellh(gline string)(bool){
5654     ghist := gsh.CmdCurrent
5655     ghist.WorkDir, _ = os.Getwd()
5656     ghist.WorkDirX = len(gsh.ChdirHistory) - 1
5657     //fmt.Printf("--D--ChdirHistory(%d)\n", len(gsh.ChdirHistory))
5658     ghist.StartAt = time.Now()
5659     rusagev1 := Getrusagev()
5660     gsh.CmdCurrent.FoundFile = []string {}
5661     fin := gsh.tgshellh(gline)
5662     rusagev2 := Getrusagev()
5663     ghist.Rusagev = RusageSubv(rusagev2, rusagev1)
5664     ghist.EndAt = time.Now()
5665     ghist.CmdLine = gline
5666     ghist.FoundFile = gsh.CmdCurrent.FoundFile
5667
5668     /* record it but not show in list by default
5669     if len(gline) == 0 {
5670         continue
5671     }
5672     if gline == "hi" || gline == "history" { // don't record it
5673         continue
5674     }
5675     */
5676     gsh.CommandHistory = append(gsh.CommandHistory, ghist)
5677     return fin
5678 }
5679 // <a name="main">Main loop</a>
5680 func script(gshCtxGiven *GshContext) (_ GshContext) {
5681     gshCtxBuf, err0 := setupGshContext()
5682     if err0 {
5683         return gshCtxBuf;
5684     }
5685     gshCtx := *gshCtxBuf
5686
5687     //fmt.Printf("--I-- GSH_HOME=%s\n", gshCtx.GshHomeDir)
5688     //resmap()
5689
5690     /*
5691     if false {
5692         gsh_getline, with_exgetline :=
5693             which("PATH", []string{"which", "gsh-getline", "-s"})
5694         if with_exgetline {
5695             gsh_getlinev[0] = toFullpath(gsh_getlinev[0])
5696             gshCtx.GetLine = toFullpath(gsh_getlinev[0])
5697         }else{
5698             fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
5699         }
5700     }
5701     */
5702
5703     ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
5704     gshCtx.CommandHistory = append(gshCtx.CommandHistory, ghist0)

```

```

5705
5706 prevline := ""
5707 skipping := false
5708 for hix := len(gshCtx.CommandHistory); ; {
5709     gline := gshCtx.getline(hix, skipping, prevline)
5710     if skipping {
5711         if strings.Index(gline, "fi") == 0 {
5712             fmt.Printf("fi\n");
5713             skipping = false;
5714         } else {
5715             //fmt.Printf("%s\n", gline);
5716         }
5717         continue
5718     }
5719     if strings.Index(gline, "if") == 0 {
5720         //fmt.Printf("--D-- if start: %s\n", gline);
5721         skipping = true;
5722         continue
5723     }
5724     if false {
5725         os.Stdout.Write([]byte("gotline:"))
5726         os.Stdout.Write([]byte(gline))
5727         os.Stdout.Write([]byte("\n"))
5728     }
5729     gline = strsubst(gshCtx, gline, true)
5730     if false {
5731         fmt.Printf("fmt.Printf %v - %v\n", gline)
5732         fmt.Printf("fmt.Printf %s - %s\n", gline)
5733         fmt.Printf("fmt.Printf %x - %x\n", gline)
5734         fmt.Printf("fmt.Printf %U - %U\n", gline)
5735         fmt.Printf("Stouut.Write -")
5736         os.Stdout.Write([]byte(gline))
5737         fmt.Printf("\n")
5738     }
5739     /*
5740     // should be cared in substitution ?
5741     if 0 < len(gline) && gline[0] == '!' {
5742         xgline, set, err := searchHistory(gshCtx, gline)
5743         if err {
5744             continue
5745         }
5746         if set {
5747             // set the line in command line editor
5748         }
5749         gline = xgline
5750     }
5751     */
5752     fin := gshCtx.gshelllh(gline)
5753     if fin {
5754         break;
5755     }
5756     prevline = gline;
5757     hix++;
5758 }
5759 return *gshCtx
5760 }
5761 func main() {
5762     gshCtxBuf := GshContext{}
5763     gsh := &gshCtxBuf
5764     argv := os.Args
5765
5766     if( isin("wss", argv) ){
5767         gj_server(argv[1:]);
5768         return;
5769     }
5770     if( isin("wsc", argv) ){
5771         gj_client(argv[1:]);
5772         return;
5773     }
5774     if 1 < len(argv) {
5775         if isin("version", argv){
5776             gsh.showVersion(argv)
5777             return
5778         }
5779         if argv[1] == "gj" {
5780             if argv[2] == "listen" { go gj_server(argv[2:]); }
5781             if argv[2] == "server" { go gj_server(argv[2:]); }
5782             if argv[2] == "serve" { go gj_server(argv[2:]); }
5783             if argv[2] == "client" { go gj_client(argv[2:]); }
5784             if argv[2] == "join" { go gj_client(argv[2:]); }
5785         }
5786         comx := isinX("-c", argv)
5787         if 0 < comx {
5788             gshCtxBuf.err := setupGshContext()
5789             gsh := &gshCtxBuf
5790             if !err {
5791                 gsh.gshellv(argv[comx+1:])
5792             }
5793             return
5794         }
5795     }
5796     if 1 < len(argv) && isin("-s", argv) {
5797     } else {
5798         gsh.showVersion(append(argv, []string{"-l", "-a"}...))
5799     }
5800     script(nil)
5801     //gshCtx := script(nil)
5802     //gshellh(gshCtx, "time")
5803 }
5804
5805 //</div></details>
5806 //<details id="gsh-todo"><summary>Considerations</summary><div class="gsh-src">
5807 // - inter gsh communication, possibly running in remote hosts -- to be remote shell
5808 // - merged histories of multiple parallel gsh sessions
5809 // - alias as a function or macro
5810 // - instant alias end environ export to the permanent > ~/.gsh/gsh-alias and gsh-environ
5811 // - retrieval PATH of files by its type
5812 // - gsh as an IME with completion using history and file names as dictionaies
5813 // - gsh a scheduler in precise time of within a millisecond
5814 // - all commands have its subucomand after "----" symbol
5815 // - filename expansion by "-find" command
5816 // - history of ext code and output of each commoand
5817 // - "script" output for each command by pty-tee or telnet-tee
5818 // - $BUILDTIN command in PATH to show the priority
5819 // - "?" symbol in the command (not as in arguments) shows help request
5820 // - searching command with wild card like: which ssh-*
5821 // - longformat prompt after long idle time (should dismiss by BS)
5822 // - customizing by building plugin and dynamically linking it
5823 // - generating syntactic element like "if" by macro expansion (like CPP) >> alias
5824 // - "!" symbol should be used for negation, don't wast it just for job control
5825 // - don't put too long output to tty, record it into GSH_HOME/session-id/comand-id.log
5826 // - making canonical form of command at the start adding quotation or white spaces
5827 // - name(a,b,c) ... use "(" and ")" to show both delimiter and realm
5828 // - name? or name! might be useful

```



```

5953 </details>
5954
5955 <details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
5956 <a name="script-src-view"></a>
5957 <span id="gsh-script-view"></span>
5958 </details>
5959
5960 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Builtin data part</summary>
5961 <a name="gsh-data-frame"></a>
5962 <span id="gsh-data-view"></span>
5963 </details>
5964
5965 </div></details>
5966 */
5967
5968 /*
5969 <div id="GshFooter0"></div>
5970 <!-- 2020-09-17 SatoxITS, visible script { -->
5971 <details><summary>GJScript</summary>
5972 <style>.gjscript { font-family:Georgia; }</style>
5973 <pre id="gjscript_1" class="gjscript"> function gjtest1(){ alert('Hello GJScript!'); }
5974 gjtest1()
5975 </pre>
5976 <script>
5977 gjs = document.getElementById('gjscript_1');
5978 //eval(gjs.innerHTML);
5979 //gjs.outerHTML = "";
5980 </script>
5981 </details><!-- ----- END-OF-VISIBLE-PART ----- } -->
5982
5983 <!--
5984 // 2020-0906 added,
5985 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
5986 https://developer.mozilla.org/en-US/docs/Web/CSS/position
5987 -->
5988 <span id="GshGrid">(^_^)</small><small>(Hit j k l h)</small></span>
5989
5990 <span id="GStat"><br>
5991 </span>
5992 <span id="GMenu" onclick="GShellMenu(this)"></span>
5993 <span id="GTop"></span>
5994 <div id="GShellPlane" onclick="showGShellPlane();"></div>
5995 <div id="RawTextViewer"></div>
5996 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
5997
5998 <style id="GshStyleDef">
5999 #LineNumbered table,tr,td {
6000 margin:0;
6001 padding:4px;
6002 spacing:0;
6003 border:12px;
6004 }
6005 textarea.LineNumber {
6006 font-size:12px;
6007 font-family:monospace,Courier New;
6008 color:#282;
6009 padding:4px;
6010 text-align:right;
6011 }
6012 textarea.LineNumbered {
6013 font-size:12px;
6014 font-family:monospace,Courier New;
6015 padding:4px;
6016 wrap:off;
6017 }
6018 #RawTextViewer{
6019 z-index:0;
6020 position:fixed; top:0px; left:0px;
6021 width:100%; xxxheight:50px; xheight:0px;
6022 overflow:auto;
6023 color:#fff; background-color:rgba(128,128,256,0.2);
6024 font-size:12px;
6025 spellcheck:false;
6026 }
6027 #RawTextViewerClose{
6028 z-index:0;
6029 position:fixed; top:-100px; left:-100px;
6030 color:#fff; background-color:rgba(128,128,256,0.2);
6031 font-size:20px; font-family:Georgia;
6032 white-space:pre;
6033 }
6034 #xxxGShellPlane{
6035 z-index:0;
6036 position:fixed; top:0px; left:0px;
6037 width:100%; height:50px;
6038 overflow:auto;
6039 color:#fff; background-color:rgba(128,128,256,0.3);
6040 font-size:12px;
6041 }
6042 #xxxGTop{
6043 z-index:9;
6044 opacity:1.0;
6045 position:fixed; top:0px; left:0px;
6046 width:320px; height:20px;
6047 color:#fff; background-color:rgba(32,32,160,0.15);
6048 color:#fff; font-size:12px;
6049 }
6050 #xxxGPos{
6051 z-index:12;
6052 position:fixed; top:0px; left:0px;
6053 opacity:1.0;
6054 width:640px; height:30px;
6055 color:#fff; background-color:rgba(0,0,0,0.2);
6056 color:#fff; font-size:12px;
6057 }
6058 #GMenu{
6059 z-index:2000;
6060 position:fixed; top:250px; left:0px;
6061 opacity:1.0;
6062 width:100px; height:100px;
6063 color:#fff;
6064 color:#fff; background-color:rgba(0,0,0,0.0);
6065 color:#fff; font-size:16px; font-family:Georgia;
6066 background-repeat:no-repeat;
6067 }
6068 #xxxGStat{
6069 z-index:8;
6070 xopacity:0.0;
6071 position:fixed; top:20px; left:0px;
6072 xwidth:640px;
6073 width:100%; height:90px;
6074 color:#fff; background-color:rgba(0,0,128,0.04);
6075 font-size:20px; font-family:Georgia;
6076 }

```

```

6077 #GLog{
6078   z-index:10;
6079   position:fixed; top:50px; left:0px;
6080   opacity:1.0;
6081   width:640px; height:60px;
6082   color:#fff; background-color:rgba(0,0,128,0.10);
6083   font-size:12px;
6084 }
6085 #GshGrid {
6086   z-index:11;
6087   xopacity:0.0;
6088   position:fixed; top:0px; left:0px;
6089   width:320px; height:30px;
6090   color:#9f9; font-size:16px;
6091 }
6092 xbody {display:none;}
6093 .gsh-link{color:green;}
6094 #gsh {border-width:1;margin:0;padding:0;}
6095 #gsh {font-family:monospace,Courier New;color:#ddf;font-size:8px;}
6096 #gsh header{height:100px;}
6097 #xgsh header{height:100px;background-image:url(GShell-Logo00.png);}
6098 #GshMenu{font-size:14pt;color:#c44;}
6099 .GshMenu{
6100   font-size:14pt;color:#2a2;padding:4px; text-align:right;
6101 }
6102 .GshMenu: hover{
6103   font-size:14pt;color:#fff;font-weight:bold;background-color:#2a2;
6104 }
6105 #GshFooter{height:100px;background-size:80px;background-repeat:no-repeat;}
6106 #gsh note{color:#000;font-size:10pt;}
6107 #gsh h2{color:#24a;font-family:Georgia;font-size:18pt;}
6108 #gsh h3{color:#24a;font-family:Georgia;font-size:16pt;}
6109 #gsh details{color:#888;background-color:#fff;font-family:monospace;}
6110 #gsh summary{font-size:16pt;color:#fff;background-color:#8af;height:30px;}
6111 #gsh summary{font-size:16pt;color:#fff;
6112   xxx-background-color:#8af;
6113   background-color:#681AD;xxx-PBlue;
6114   height:30px;
6115 }
6116 #gsh pre{font-size:11pt;color:#223;background-color:#ffffff;}
6117 #gsh a{color:#24a;}
6118 #gsh a[name]{color:#24a;font-size:16pt;}
6119 #gsh .gsh-src{white-space:pre;font-family:monospace,Courier New;font-size:11pt;}
6120 #gsh .gsh-src{background-color:#ffffff;color:#223;}
6121 #gsh-src-src{spellcheck:false}
6122 #SrcTextarea{white-space:pre;font-family:Courier New;font-size:10pt;}
6123 #SrcTextarea{background-color:#ffffff;color:#223;}
6124 .gsh-code {white-space:pre;font-family:Courier New !important;}
6125 .gsh-code {color:#024;font-size:11pt; background-color:#fafaff;}
6126 .gsh-golang-data {display:none;}
6127 #gsh-WinId {color:#000;font-size:14pt;}
6128
6129 .gsh-document {font-size:11pt;background-color:#fff;font-family:Georgia;}
6130 .gsh-document {color:#000;background-color:#fff !important;}
6131 .gsh-document > h2{color:#000;background-color:#fff !important;}
6132 .gsh-document details{color:#000;background-color:#fff;font-family:Georgia;}
6133 .gsh-document p{max-width:550pt;color:#000;background-color:#fff;font-family:Georgia;}
6134 .gsh-document address{width:500pt;color:#000;background-color:#fff;font-family:Georgia;}
6135
6136 @media print {
6137   #gsh pre{font-size:11pt !important;}
6138 }
6139 </style>
6140
6141 <!--
6142 // Logo image should be drawn by JavaScript from a meta-font.
6143 // CSS seems not follow line-split URL
6144 -->
6145 <script id="gsh-data">
6146 //GsellLogo="QR-ITS-more.jp.png"
6147 GsellLogo="data:image/png;base64,\
6148 LV0RwK0GooAAANSHUeUpAAQEAAB/CAYAAADvs3f4AAAAAXNSR0IArs4c6QAAAHlwHlwm\
6149 TU0AKGAANzBAEAAUAARAAAPPEBAUAUAAAABAAARqEoAMAAABAAITAAIdpAAQAAAAB\
6150 AAAATgAAAAAABIAAAAQAAAEgAAABAAOAgQADAAAQAABACAgAEAAAAQAAGGawAe\
6151 AAAAQAAAH8AAAAAYx1BhgAAAAALwSFLzAAALeWAAACMBAJcGAAAF3RJRUFUEAHTnQUFWWZ\
6152 ++t+TukZ3icGg0/yj6Osb8Wgm2Avn7uG4+BISTRYnQXdkPQcK2j2nWLD2MSlRkEUAPOnd\
6153 4iuXJ7jrIY250D0Gmf2Vq1BEISggCoiMMA+mu+vu//ZMD9U1daU6a2UbuV91KRq3vvdX6/q\
6154 fnXvxdStBA8SIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6155 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6156 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6157 IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
6158 2eXs9H9+ftSkSDHxsc2qddE7YusS+1qaaLKfnY5YsokMHWEPtdK4MQPz5UEeXbLbYsAU15\
6159 npd1LKXE2CfLRM53JSUaq9ScqcU6i+2kK3StuOnY5reEGKJ7Qw7mOvKec2Toq0iZwo1jhfS\
6160 jboVHCstMRb3USXEJ8hFu7DsdfB2+uX4VWVFXbBpMeZULAE/hcKOGab6eKGOlNykH56PC\
6161 HxH2VV8KRqgh3qUeKilYdaOfONJ560kd16w5BwomnOQlyPziON9DLMxPFK/60p2P/Piyovf\
6162 N8mfM+nJWNgnjw9KqOTOLVGSFT2p2R11gn31j0VK7YsoWmZzEuVFPfLRKYdfoakZLRSB0q\
6163 zrWocCOG6gehvgRaCj/dktj3gdXXH4gKN6AR50zpzYzerq6SR6aZDQqfK79SKTRXhu+e9/PN\
6164 L66as8BpU/PNlpN1TLJKSc73dPXSr20ur71lwPccBqhbNnCyHu11ryy07QYpF5FvgB7jx\
6165 +chBjBj59pRydlJHUy3984d40H2Qtx8THaPeFI0Uw1c+KnyH5FGEvUWGaeX83SeMKOLf\
6166 r1kbD9HfPE5VvQ14h9PFUkkyfFbq0bnzLzJq4FiesnDHcWUoeVQ0k/5C9F9P1Uue0H\
6167 +zGhU9nsGqrmzUgaurk19RpjBD4Y6uQcQd5TU0W6z3D3MHeey14V491sbdkYxbGHLCFR\
6168 UJ6toACF7F9V58NBFDHTOMBaE74Ent+eWrrWr+Lz/QTW60AdB7JUjps/OA7COBNCeMUZ\
6169 ttCu/cOg28FlpvKELTPVf8JruasEahhBvXaR1guoeBPyfUD04+OfeBdyb8L4z29eSXPfMOC\
6170 bgGogov0lzgGw4jP392xnHhdC+Mwf3JTjftnt22yCIYJBJXMuT5KIKyck1xSRd1d6BncevW\
6171 aJovvY/VbacMevgEP46/Zlnjt9xj17VL53215Mtvap1QGLNHw5P0dqXyNTQ1z2b8nCGM2ZV\
6172 qOoFjSjYvV0AZzdfayidv6FJ35CS4jXZk9hir7e27zm6p3T8hLJpkYicJpV1HtK/DJFU4Jw\
6173 1ImhxM5IR9vZzgrKx4w/C+HQSPe+krblyrN3qEPTNahsHaldS2xh5Q5NCoPPVdEgpcqbm/8e\
6174 7/zdoOaptag/mlKJ77U0VG0xybTdx/Ex/Ptfa/i7r7Ku+cSoiCkUwrohUxFl6wEV9H+ccVg\
6175 pd/CFU42AKIUIP1vTKL/sjYjE5PVHq728NzvfUzvvDODGy9GooouhmlNfctX48YHL2qH\
6176 f/8hpXVU/43rOg9xtq6Ytcv1XDC3fmdWQn9nbf21e7wKE1b0K65icBu0Eghd3IaW82dwKPu\
6177 hrauc62CwdkCjUZK8EUXMae71zUqwc2nbi6eVn1Ji9/P7eW+loMaogF+NI3jJLSf8dn9ip\
6178 WNW4P9y9JxuPeDL/HXzNzTsvesLD2vswHwI9mu5rvVzX9foS4v/LfmqDEIphDGLfM2uCW\
6179 gJTy2wOENPa23fEicvd+2YNcNjYrNyhyGao8jRoJTAUmRiqOCJrRW5FpTN++frTwdh4SiU\
6180 bVlWbfffLcRF04qazRD7176/rBjKYLd5pB125W14wQu7t1kPBEcOpuW+Kj0sqP8GHNoAZuwL\
6181 IO2uywDh9zBz2xodrQvQF15QxxH60wJrKAAW46pvt+RxAJVLjW7Y9/+CeUBMK168/rPQn\
6182 mcufKzALdFN/yI9gA5lWc3dkIKhsyVzUCfSVG/KHchwPWRKAMdDBEX++HF12A9ct2d172\
6183 zqWzOvCOYMEtByl7QogDXWAAlQ7cOQZchyADWnerGNSXVtctj3sdSp20twcmkUJ78+2H7\
6184 yhYbUgm1X7f7K1DwaRyUFN42FtuxNdVdvtAm66Yc9R26VtZwN2px8NfmeHz3EM+mgsoLk\
6185 d3/ZnBGE1XPGUWzXglYc45eW5/zBcy54aAQWqKfNwbqptcevWT4FUBvov32gew8LDzMTMa\
6186 aupp7t/bMXX+yw/egJGKotksy2d+gFbb9VodvX5B1ZTOR+wfjy0p6U0XGOYnqr/quta3vB\
6187 Fgeua6qv27vn8dFdv3rldBw34GSPg910DG9H5Xknh9kAAmYJ6dkLp2ZtD3cnu7vtv5c\
6188 h/YrG1p7Wxp/VvuRDuc+wsq54ymm+8z2K0GyRSPrA4IKoGz1i8b6ytagcEPmb9/m09UATz\
6189 Jow6VnPmXhZj+sNnPHsCjYja6cRsMyrGkiwF4I5UoiuLiL1RW7fmLNeX3z2+/GfW1LU2\
6190 Y572b6EazkYoPctJi15Q1nJyLdrFrU2p1/3pmkuG/yN9gAoGYMTf7neV1v/6CHUgk1uh/\
6191 F9Uvo+gG703q7rzFl8xq+zW+/8FP6W6FV7xSxhiNlaywWdz2X1ULm/4uLmPwNoA5uGcdL9\
6192 2FA6cogozh7G6Q41NR5Dj9xuvlcy+rFbcujVsnLkKv0CefphUbICLRMv1+9KP4vngH6Fc2\
6193 NCQMSiCsnCkfxed+mtFlBwuxdmfB0zqt7/L94225Y3TCzrpQWhthG2zHraJO/yb0kdhpanz\
6194 GxWf66/8Cb5AHcbzdpnhUeG6YFow1gzeMmtGcNdekzTiXVuc3Lk4yVTJepug5tgSWFKXd\
6195 uF9MfwiG3sqnTcx76+3eXQWwzVeqSpvr2mC2afYsvY461+04KvyVgieCugG2r30pYPTve\
6196 o2Ulm2JWE2o+f6K0dPtNXf2U9x70/bqzct5z0Poio+vdpyDjcdxrd349UxCeHrLoSktt3ug\
6197 AcwtK0O9FzFn+gWtWds60dcFodrAxneOCFRXWUSoK93pB2Xn7vAe+qwr506/204LXngLbrC\
6198 76HGRdvtHz2WlMYVvqgm5zTP2P5+7voLRRzJ10Y1x+BoH0eZb+CV/0TU5ic3NGjks30M2\
6199 fFUtl1+y4yfAcwKjzqz2yb6HlGjebwpgLXoo09/j8k//WW3xS32gQPHrV5AmTP1IDFN2Op6\
6200 fz5yWf4HfMDX/Buy4NVU73yEfbOR65icot+zjP+8qf4JKYiTNgrTB/qST0zMKACq18jPGL\

```

6201 A4PCxYnPKotjREv84HpyOsws/BsqyT2RGZ6zr10gA9sBHEp46hsP2ratm0JeGrugWBDB2Pw\
6202 NY01B4OSTMmcmcdS2E/GG2ZvrF7Uejsqyw/7A7guEH6Kyy19q3fpQvQXTx4dz+Ueg+Lmy5v\
6203 bjJyO+b5LsgpG5Nz6bnwBfHduAYgemz4ap1z5d1bByA3NQTC4F3RKYfOTKaUF9XryDLwU8\
6204 sDMG/H9v0GTVNVC+zh2tuZ7rgeAbk+48H3P553q00yu/WHj21ZmBz7z2KLWaf41gm0SV\
6205 2GML+6Kmh0zrvsQWgnell1yZ/gLLx+1BNcnZQF7P9Y5XQFN/qIa+Hr3rAg1MTLR33bfyTf\
6206 m645oyCzZJmXznQ2JAgqbBymKSL9VzQsGbfXtjHbJkz2m+VkuERBRI0E/Bw8ogf/LI2hY\
6207 /9Tcnsh681t7DtdgnQR81EvT29eWt58jF71FSZ0VlyTfLvtUOTb62etccbr01HeS68SvY\
6208 20zUdeqWmRW7S7ng7dKrV9rLztoMPBK73naA4Yrd2fz5d5SzymymaInC1okvPOVHG5F+Qs\
6209 wCv6RwU9Dkx5MU9wQXMaX+ePguLw8/dvfg6U1LPvsFPpXspOniQwagE1smgqXctOE01vJ5\
6210 7tBBjJadHkMPdY0/q/r1w1b44t5eCNKQWAg7DsuJzH16C21k8bk+1u2u78FYXWfkLQ4/QY2x\
6211 1YvJx8boyWm6zwc9/Ojwz7pUtv1Lp0N02UxL08PKODMuluov0TDjLyxorNWHEHjQWsyKrkPs\
6212 2JH14LpJicXoyp6mS5fYsKeile0G95+Wx0eJ3m5mcmjNe5b+lyH2YELXGjRmDnY/HtmK0S\
6213 apE7Md34PueUyZ8DWDovSjzXVF/xsFe+Lpz/wjQQ9eIh94ZwqVS62+CUhV3LmTnJSHfXorHF\
6214 wKgz9PwtrCRJrJwh5+/ocSLzQ1zG52BvITg+WopqXRYEwcaRfddbSg5CbD/PySxhBAKPO\
6215 qz2y94110uAB4x5we8qDsH06++b0nwjzFXyAUvIy6Ece00117SA2kx0UgxtmZB9CAVyx\
6216 2CMBJAdTCrWwYKriwy4myT9z2t3R93/8X1J0ESWetty7qF1J0dwkAmhFEA2ZD601wN66h\
6217 H52HwW1ALQHQQYzwr6yznTls7rgu40YBJq4BJWJCayRHrTyeYx4X8/xCw+rue9155yc50A+W\
6218 8v0W0N2Zxw7VADPZcEdpXpdsLXoDKefrWEM+yj47aEAA7yxzHjXm+61FzUL46ch7e0d6Q/m\
6219 Wncf9BvTxb623hNxpIvmlKjhJubTFKRbaG1QCWibwui1PtyKlhwZag8YKoeMej19Y9LY\
6220 PwK79U55Bk75f5XmChwhj79J35x77q8t89pvtBqSG555hdj1n6Y8ErfyqVOL2XoeLrbmWj\
6221 WkqG5S2p1IOK5dJzgg8ZLB1B446/gfucosafyuW0Y1zCuoG41lqXQOQep1w1XUL4PPR\
6222 zB3G16m3A35xPk1NLSubb/3kcv86JXGzrflBBjBhJ71lmbCDRVdb1e1etqPbH\
6223 NQ73iqMhz7ETHvurXm45r8FPQRWnD1qfV2zqBlxFl6+rqDLV82CtnVYB1d8s2JfEbwHJP\
6224 aW3rYbgn9qXmLnChjcnvUN5FKMRc2LbzJbk8m55en4x/2rLdJQzNjtkyuu01pdgceFM\
6225 gKgp/ahfXooVi+JtofImZuJyn87HQHmAmXdaUeTX6c7F07sUukyg50z33v/20C7b+scH\
6226 LtntlH3YeW84ipGT4JWAnu7P5xwqjxbA4Ibab3Q8rflzPCf7c0Sf0B8NabzSfWqfHbU\
6227 nmlidJITHGh3eSRt+42Mk5KwCtsFmX35RJTvorP3rnm49VM0gfP8oid191X61dvbXmkqjvb\
6228 Nfyd9X8mWim2MLKzESL/VzQSKDPzcdYcYTE7lq/B4XKfQaNeK3mL4r29FQL/gaT+vrEO\
6229 gDTT0XU9UWkUVMfh9MYuL2jVpZxxu0FP00/pTedhod/1XXxGZawfuXp6E5edi+eme2X91bo\
6230 0xuU119PbLaKGGhafa5NVPhxjK7X0gLuOMRm+JAFesnnaKzLRhZXLyBf5ediUwKc1/wD7\
6231 dF+JL72vTEPEIqgkz76zFP/d5duzt+2HihxfKlnhs7umT011AjKkyVScenp1W1AAACzAE\
6232 dqV2Sx/S+nL0dPelXVtd/skUr+JL5/9VsbL75z+bYNS8Q2EuQN/Oa3x1/FJZS/VZ30EGcB\
6233 ePdtCYCR0Kcr3q6Vl0p0f7XfXvDaaVzc0jQECZ56CyYcmxZ/7CyWar2IIN2XK4NOCO75/\
6234 4yMTRk3XuyfGJgmxT/xdbpt8uSri7F1l1u0FJtmJ017cKfYqMVsfdvwpV9R9Aeh07FR\
6235 hUL4693puuYn+FX0C+CyOvR1WxZylh/w3n7F1ie1buteTsvURMitjpkWRyMkKzMHdZFeiM\
6236 dM1f6+eW10/651MmDD2YFEL2dFycyGj38aRABqSGPX1sCGUCaKRD0UyszauvgcZx6zAVTf\
6237 LLGg1XPJfjytchKphr+cN+76LoLJd3d451+sndv9Yr4veCw9+SxTcx6G/areZLB4Wk\
6238 tgzv7W4m+Z8f/FFzZUKIa3ky5ULmo9CE8N3HqLm15sRBy3ZhsXooRnTbmbWmIn9T03\
6239 j0g8vnlN3ZecfFY1qcm12/FviCj0XytlieoLL0xvRghMyN21/It16w335y9/711dy057\
6240 xLJdUm+X0PqgtrucgEUTDVIpPenovW4f2KAeVArG5T3tjBQ0t+5rCIU+I1BzPPIPJumpRVP\
6241 4YEuz9wP9x1fvw/0ppuyDp9uNpYih19/XNkovNSd5dGG8Cwms31CzfrkCQUTZSjz+hbm8q\
6242 JV7E3Xm6WjLSr6LVB68T0ExtHj7/4Cdw24+uzFvsJrT11RkFOOALtznFzdf2SD120rQ\
6243 8Y8V88Dsb0vRLQD6exvrEOj9y4gDQPKC5ZmjJyz021ldV7yb3zfl8qsmDmOFARVWFC3i\
6244 N1NQGW1JEavG0MrZ78D2ZVeHmcdFCf8C86nBB5KF1FMRHE6F080AtOm/d8V8km7D\
6245 C58YrseFvLvsPlpbx79z64erdZnyuNLkileJdalUak7J0orr315x+YA9CBQBDf/cK7JkHdB\
6246 E5sg690KMhpdRjdv63vBvYbdQucucSLVM9no/QaPP3KZ1ve8zCmCj30kX+30RK0E8kIW\
6247 lBxaFhe2JqBL8of8Gkam6n5P9mdGp5bmUikpmc22LR7BHSKjJP0kmCtCf/KAMsOJXtje\
6248 v7g+/OzmZbn/25IOHT3+NPgzn2eyx7u0iZQDM9xoyTCZBTOya+vndgW3URP1jYxmbDoe1/au\
6249 zq4BtYggs1mphGdLkXcmLwXskzBQwa940stveb+f7141k3oi05Mxod+r/1ZvX0P9Ec3\
6250 xp7XUy8JUGtqmatO+NeY/99v3xhh+21bh03cn0t1jdfCZnzkeapSDN/vJdg4XPnB8+W9p\
6251 9z2duKz2Q3fev05lytqto30pzK9Ecs5HOY+FXFV1506rX5HkDFMGaakQ3YA09YdfJdY\
6252 ppf5kjqNq6rny13dfYK1H14o0Kj1aZehB9JNWTfBAGVvu1uIawS2XvTahf8S50dfRpeEaP\
6253 mRfLX0m8X4mnp/FBy6aVg2f2y55Kwno2mMPFSF3sgCf3o4UGGSj/w1548wLflvBvab7Z0B\
6254 Xx/Hw61F9ZrXPQmBx5C1AfiJiHjYjhsR7BKMfG8mT+D3C4J27go01VNN3j3d6Xw7hyf\
6255 xovSf0m1d70z93h0LHBY1REh77p1e9XWk13Jd56103\
6256 iBWEY1G5BEaJ2NR0adga7IEVORZLBS-CVC820u5UelJhspVgHEcusjRkKYLW0SSU51mW\
6257 LaycFxpHsw1kQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
6258 QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
6259 QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
6260 QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
6261 QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
6262 QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
6263 m38W0ncAAAAASUVORK5CYII=";


```

6449 "UIRjLYAaUsPBKMoRnqltRwTgKXQeb8P7Jdhzr7MHSHFDEhduNVL5Ew4wJHXztyvYw96Y"+
6450 "pGx4PzWc4xBAmI/xQ009CAIxYu28BFw6NwZzplYhLSlyEftZKLJ5UzHpwdfZFRCC271dAn"+
6451 "YmPCKBQbkbav2Gn24P5Yl28cAOWz6qwyVlHvHrEiUoCkHXEdSC0B0hWhrrBy/AGT5vhwz"+
6452 "oKkHXg7l5em4cFmsYlNgAMnB6QgrFyPo2FE6FhJipom6e6bPjotEuz7YDmGdIKIEAB"+
6453 "pBU7kqkP4S1emTYzE10xY5MkFTYzok0djjxv7Sx5G045qPH+kCuTQhJltpoa04v1wCh"+
6454 "iSDT6e5e7GzXANHFkhh15S0yHt+H/051C/sPlmXFw3Y7dX9jEkaEuLhA1hBNC8zVAkXzH"+
6455 "Skd6aPlmGtiTsetDMQqokzWuz47AgsJ4LJ2TlgnUqnhqiaClVQ7DpeG01vLcajevJLJYXBt"+
6456 "muorKHWdAg6H6G6F6W640c352HR+eLmXDSX5x5wBsfVGPz/KUUT/VgXgVNNJprLmMvRtd"+
6457 "B1pGX1GXfcUCh0QoIYaZwYj7unpxuqMx91erM2LwvL07FsfjEzq3vVpFfT+XYofv10L"+
6458 "5YmhnIYodUMY9TusjEa6A1Cgkbedc/ctTsX666diQrJfXJyGfmaZv0NHzYgk+qcy/kZwroy"+
6459 "abKMMa4EFDn2KDNLI11eQ7SCDN1Ir3qU1X20bziuzseLITMRE004Tdu4/jft/dxxEamyD"+
6460 "tGhYfGjICrn3pgHw6Oov15RMXQTESF13z2TgdunoT1NlXUaVez08uClv57C+18fRU11BzW"+
6461 "PiZyX3sSkwXbiIFOctqjnyH2QNeK49msKDIVddSV9EacBmB4XCuF9FG896k80Ij8z2jb"+
6462 "3KxYXDocQCoqOn+IpoRadAgIYeg+RYPD6UR3pwtBBSdmsSs+bbvg1XVtCC0Kyc2K6qxX9t"+
6463 "PIQfba/BRVS9pEh6VuRcXaadChZa2tKD712cgulXUJjveCtS2T5Y3W3UtoQcfmEPBHKU2btxd"+
6464 "14nDRHrMi1f8gpI5K5gVb8HhL23IGEBIoXKAFIT3jD/vypcFryQ2Xj79FA91n.6BCdq03Ph"+
6465 "Mo0PPtQ4GhmfZordTixYf1MzCtKxubdNadiKeAg7q1BRFLfYCFz2vPYTnoktFMDCS00yGV0"+
6466 "8exU1NAQ7yngQDqFIKER7QAY1QBk7d+kn7b1BwVYcV6m6eC9fdw48rdfTUcbitXTXNtLCAj"+
6467 "+taZbjy3n9zpyUjhs/RxBgnhtT20ZVx14WjEhds61BfzjoQktVC+ZPTzCuHmCk6U0NGk"+
6468 "pNop4YvwwB8A017B4FYsgH148VbJmGVb+R30dCse0Y15S5HIXWSU6I1E6eD0gokN9QSkE3"+
6469 "nrmWIK10Lz2V0b3X9E51xgkitY19+k0MCG6vBk/seeFgm17KRbbngfVnI65SH0cixkDW5g"+
6470 "30MQR67h3bamdCfWuaobvAksuevYllk80Hz0y9sFPjVR1401b056/Lg/hpK1vY1KX71Nm"+
6471 "d5aJNChh/aNesL5qF4gbgZ7/tukDUGUS1Re0S2qpTRNR+StubYbV4KBJoJaf706a4dVWO"+
6472 "z3PHEsOkdhX1Z+rcCN2L7XjQdX260Q3q8M2Ed01eXqoQ97VrCFMCD1DI9xod740pdE"+
6473 "wuzC2Pzspkn4yzD9dHKAVPA9enj/RidnAn35104ePnH08b8RcnKcjJhBSa0hjdwt3fr1Bhc"+
6474 "e9kKX7E/k9a3qm06qgrNWE99FMTvpsejrzVumEH7ZxG6qUT8S+zE3C61bu/cgk2AI2Y1/L0"+
6475 "yKIULKaHJ9iyowzPFLCgk1vasc550wF/FBP4K0X/3xdFmJp9Ho1ffz4173K9E51VfdtXgSy"+
6476 "zCs5VX+8uRRVZ9g57Q7BumtyUUXhUIChdNsvAIOPfjgrRlmZ9y3bo/TwhTtaiatoY0S8gIrs"+
6477 "gx2wkK0GuxK4vh1g6sYTL2dg3qU9gW2MAoqtMUR3XVRm2D9W+/F00RvqmjLUG7Y20MKWym"+
6478 "oyY4ryA9yXwF6ktrB2m5KxZs6B/hy7u4q5uWm2Dev/xuQMLGi01sAzWgZdXc4+9vdh"+
6479 "675I/HsAC3NisCtV+SaKpzuA0bfv539TnU3f1yPV050kmiG0No30cAS33LaAm3iwm2Mk7K"+
6480 "4zvYOYFGWZ7fs3SbZ7Qb2xORZP1E4RYO3LOH94d8UCa70v0d6Ip+huzucaq81FB/3Kooow"+
6481 "lNyabq622324TfVvdniIXwU1BxseHG87MHFDH70fyVh1t1f/89z1wtbrNnsLWwjuSE4hL"+
6482 "5TKMxARwx+Y0Hy6wRNJLcUcmLrkojgClA2tsvTMjvP6rFP04eBUAYAdIRRwOXznooinyJ"+
6483 "vr3D04hmxxKsVehqZ2n2rGegYwZU5NN4/H8BGP67sWotAEGEDXG0dJ4jrsLGHtTtdj2gd4"+
6484 "Kxc+1w9zX15hXN2qZPn8CHT.xdhwL2S31by6DUfX6F8XhCs15d2Q4Xj/Aj0b9yZ2+gh0H"+
6485 "13Py/01uHctw1Snj0y1h9no3L/qmFLKwX/sYQWu49PvP0E/0bJyCLAYvabfqtZTRd7"+
6486 "e06mZEl1TGSGSLRxy08e9utpJPSWCDokS491m2ysd05ET2FpwmL107atPkYbS01eMUS"+
6487 "Z690021UP6rVzQIRNok06bGKRYAZuI23B6K708X0Kna14g2UMjffTpehZPd1E9Mt"+
6488 "lFump8Ezowav+VogG6dhuf+dChPzEQvT9XWZLJ3M1YmkSbaPQHDWai01PvnpugLXfvJqZ"+
6489 "nAYbhuCOGD1/qcYvr6gIM4Yt0XGLItP.SHZSHJgZ2rdwx7nMH31Ksn4JYShk0cQf8dm"+
6490 "F07hFlPvPqqaBJtaAsAVMdb/keA6uCyYwdLGF0Waf7p6RQ7gamZFmjrdm3lg3CV1M0Z8J"+
6491 "diga+4fjbcZJMz2a2Wbdnbt1k15aqJFNA3nost7ncVB+K4y5CXXRFRFTVY9w9GgpTK011"+
6492 "T4i3Iq8tr160UyBqbg2Wp8WetNlWcePxo0UvOHyq2UHVbBcVgWCoARCLRpE12gadMSEFYt"+
6493 "0tPBUE62enk2YEGJaFqoFRQAKEPnxpXcgCadiTCcUeNua0j1wHud04n9fk4VbkiR8501GiwC"+
6494 "wincosPa01anVtUkgMqWbOwrK7BzeY2+QRXAgqesPm7jZB0FZ5rdSGFLYL2N7bK57P80yZs"+
6495 "21tp2LoLj+/2rP12Iy2cwi0a7Kv60JrQoTbcLY/pXG30ZampCeRX22Rjvz0qHGcWentdHRU"+
6496 "eVj9NTW8H206vtH1/01rZVCUMbmbGq5U84DC7HFO5RRAITo1hgptHrUvyOEMdoFmQERBU"+
6497 "QmFsgwVR48NSEUEkK7m3vtChpPC1h41kG4hF04g2mqTgRhmLkjt4F0j/QIG6Gablswsk"+
6498 "JfzqP3nMUaOHlqSgoVzJp243F6TMRP/IhYgnAJP3CLhtGABLPvtX+gY7C5Q87qf105zxo"+
6499 "Z6xdkPwt0QPf+91rZEMj7x+o8a9U4m/pPFMKpFUCDjzCLRqCKGoaigUeakyEbb9cROB58"+
6500 "N05Q6E1hs884pPhs/jQFEMmao0/9NTXwgetyMDkzLnfnVQUsjClwNrtfEanMtGw4vHA9"+
6501 "1yoqKkLx66zTquCnpqOxUtLecJTWnz12mfnrRJSBJVY7jeWk6wUg19hX88Lx5Lznoq3n2"+
6502 "me8nyC113KJBTASGuwrVzpoWJldfCw9HMLNJTU+PplGCoIPAM7fMfmuHJ4doErt16mwkV"+
6503 "7PC06/FK0168e06/681NBUwBaAK6d00A8rJhNtsoyvt+20JX0UT83vt7BspwXf1n"+
6504 "umTzrF3Qz3uhzbf+o0k8d4melj6nS2hMLkKh1h7MC657N83ealABRrTpe21AgLS+n"+
6505 "DzA7P868r7jxog7xogJQE7EM6Nt465GfHxmF0VWZMmeUe2iwr7j3k4dBT7/joPChzRm19"+
6506 "GhqpX70zeGvhlmlnaLpceCioBrS1gnc30VnWwIoq3LhA4b/ImUEY790xvZ23cr1tlasG8m"+
6507 "MkpoFY05f9A9er/Vi3+9Mc94juda9f1p0b2ZCmRAKzxoep/5SYG2QHUKmsooNjYlGCKS1sw"+
6508 "mk60tN840oqShuCXlOKAgplUq+Wcbp301NSBQEFt+q78bPLONdxWnG470tP8Igr1P1Tj"+
6509 "oqdoLLDc8769hibrJhpFq3iirVrRjnJuzezDjji9i+brGgXWJDGWNpGk2N3j1QbwLmWnR"+
6510 "+068Bz3MnzvPswN5mNJVIRuXz3F9C3B2Mxk/22kgf/B3GfW1/NQBvQWVG6VW0YlntBoWkz"+
6511 "SgwLGRFts50M2hKjJhTzW6j6H6S5mKcR3MJV2YPMhF9bXFKHSasW20C92PnL1wDmJDX"+
6512 "1z27+5vJRvMjrlKhCfxnEUFEDHLL0SUnFrdfwXDU7Baac4Cv9lyFMFLW42zo2Lh1L20AX9"+
6513 "eA0zHG3RqL7uZrZ/byFKSaSExb99eiImZL19azEYDvHLAaJd9iNFmCGJfIh/ICdhZPLYu"+
6514 "SOTOibVp8903yfnRtdjYObYIyFD7b04RpeQ1YRBdW1ZLcqmG/+CLWgzvIHzmXHSwvfqj"+
6515 "xwRBw4DH+FDs0AqXcT/69dml1PXYyG8SwQLzE2b7HcnwtAe1tpkEbW0TP7KnH6qVNmJ6X"+
6516 "1jTW50C3+xxrYMUW5L2Rgnan9giYfvkYgPLrHfvyZPpcDz5Uil+zT6T4sNNF7b960tBBFY"+
6517 "S0NT3eU2YdyHsDp27xY9YRHEIirH7+d16Q18XPq/TwPbubplav83zAqQ4dPOH2ibXzpu"+
6518 "HYjhi01/LacjBzN1cd/rj61vzChwey821mkLGENv1Ygs+zohEam7Knu/DumG5yALNARlmq"+
6519 "qzZ0xqjPULj6y8SbGanc23GoMeaVZNMx18eacAGngcUUPsVBbehXWASBbJCuouP8Hj509P"+
6520 "mzXXpgd6G7qegF7bz6m9pCbB9LXldw7//7AW9UQ1T9e9ptz0L3K8E51VfdtXgxi890GDX64"+
6521 "J0+qgFm0/errSextYeff7zef4AAXCahYEV1+Lk6jhYPSOH74UKKhf70mPzok/j3g19"+
6522 "KsPT//2nUhn0S7w5yljpx0eYvYvMdwEaEYJLJog7vUan3650qXzMBw7d/YWY6nv2B0rWw"+
6523 "A4/3B7FD/quaP1tCoPZV44ZdcljGPKdXbhYKtyextJzGBNdrLmpLa575yYeUuel18H1kX"+
6524 "TneqV7qjhXUQkvbfu40H61xvuzafNFPZ4cb08r5bhqMdpKzocKGAh6I/4QrwrRaTpp5"+
6525 "MKX+a417GxUjWxyeMaK51S6yup3j/eUOh7jF66qWxYpHv6d14VN+f+e/PscA9a6chjkjB"+
6526 "1XtEtjH2K1Rwd5hhSAbkCzEps5/5LZxh7jxx0XID13HW/fXYRRdDmV49WmyBCHRScLPFZ"+
6527 "r50w670W/cq07uxClW2QyO/q7Yf8tEq/tvn2GS0JWR/vdzH1D05GYTj9Mx0MKwAw5xRE"+
6528 "uColFLe8CbZbfekmyzivtWzEl3M85Nh1JyTWELySCTaGh/+1EJd5S5bS7d982f0m+P4W4"+
6529 "tAH8c6Ubd66VgP9cN8uk2yTL7NhdD12XD0yyGDUeKbM16PcXGEs5d4m37/d/km30BkEQ3"+
6530 "6e0Y0c0BjPnInUeW8q6lMnL/152pjFIUTRj+6rp47dxZB9fMxnLMSxotIzbgJ//WgrpnIA"+
6531 "25kAwSgVcf3VUDQVFlpjrLXcn169A1X0fxQd6bnJyGfYBN+GHEBEO/Jmpacr7ncbjKTKz"+
6532 "fivoALz+DIJdr/Sf1+5HA4ZxDrmm0Jh1xXF4ec/mI41JmPdAaX03PmCQ5UtoCOP3p3qDdel"+
6533 "n8gVAAZYaJvVicr/s78JnjoX02jirdgBj9SvmJ2EUHC3X7aZnx/HawgT1fhjQRi3AhihEpi"+
6534 "WlFseaFKnx+noQzdmK3Y0U01JONL5PnFAJlhN39y8No4xP409HGkntV8WQ0NpoMr+IBdT"+
6535 "4eg8p9jcx1S6xMzZ1oZouPNUlNPVxmewFtry/gkpZCzRHxHAzLpdY/+zqk5BvmPnRnMjX"+
6536 "01Kk5YyXc37660MU94ZJdp8L7BUXRRR0gVXmtGd3791Evdur8Ug7h0UMZyv4+LSVQad3XyY"+
6537 "eWBU0aC0Z2MIP4HXJuuKupp1a726zdrPhorJEp4v7Qd+5nllc190BwNps7oot1ey6hg"+
6538 "V6A5a3kblfHb0U4j425K46ajkealKSX8YkThr7hobswILYkik3K3ga0VzA2z78a5mLE"+
6539 "YoePi1CVPskdogi912ml+maoK1DGgWBDh/KFN8M1Lzkn0M0Q1Jlhqgd6f8Mv4E6fyQVWw"+
6540 "v/3N0DR+Mz5V3wzplzPZGxsaGL3+L1LAL+60wv8da8M5gmgqXnsBrfjCISZA2cThJQWz4"+
6541 "FfdvRTE33atvnrKumZyL8t0WqKupjByVhdHb1+L4as06jwhn1egq3d0nPxuFyft2Bzqu"+
6542 "kgC2fPyH1XjQWwtbdk3+dvYH3W+G5N8Y1L1VyaXpi06vye7Zf085tnfDwbgu8Ew410/zA"+
6543 "Lsyzf1EebmdrGzVnvor7bpA2ahUmrjykdY+mq655Lm81vg6NiAYQs1TnGzBNXQeCkCPYwQ"+
6544 "XcB0tRxn+T00NqITevj1aCy/HOWVwxshFv7n4maojOpqPffjJHoYx2Yy89zguSFHO44r"+
6545 "174GvWY5SMR6icAR0sMpa09P5U1tr3n6ed9DAZBWTCThytOtn6MkxuIKpt+UtUIMzxxq"+
6546 "tCywrgEjXGWBXBJRi6uvMtYgsurU03AiqotXG32S8K/CTC8mo689CT5FbU+JL2Ayikk+g8E"+
6547 "+fT2e7tu41/VVLEn8ShhNF4R/BKiajfyDYbIVVTV6DhXs0yKbXWJv65JfDY+zroO/USKAY"+
6548 "Xwcb/s3LIX6sPrcRCP1v3UH1Crtyps/n5/BCUFmtF1Hbf5/P/D941z5t1uE3A3AAAALFTkSu"+
6549 "QmC";
6550 </script>
6551
6552 <div id="GJFactory_1" class="xxxGJFactory" style=""></div>
6553 <!--
6554 https://developer.mozilla.org/en-US/docs/Web/CSS/line-height
6555 -->
6556 <style>
6557 .GJFactory{
6558   resize:both; overflow:scroll;
6559   position:static;
6560   border:1.2px dashed #282; xborder-radius:2px;
6561   margin:0px; padding:10px !important;
6562   width:340px; height:340px;
6563   flex-wrap: wrap;
6564   color:#fff; background-color:rgba(0,0,0,0.0);
6565   line-height:0.0;
6566   xxxcolor:#22a !important;
6567   text-shadow:2px 2px #ddf;
6568 }
6569 .GJFactory h1,h2,h3,h4 {
6570   xxxcolor:#22a !important;
6571 }
6572 xxxinput {

```

```

6573     border:1px dashed #0f0; border-radius:0px;
6574 }
6575 .GJWin:hover{
6576     color:#df8 !important;
6577     background-color:rgba(32,32,160,0.8) !important;
6578     line-height:0.0;
6579 }
6580 .GJWin:active{
6581     color:#df8 !important;
6582     background-color:rgba(224,32,32,0.8) !important;
6583     line-height:0.0;
6584 }
6585 .GJWin:focus{
6586     color:#df8 !important;
6587     background-color:rgba(32,32,32,1.0) !important;
6588     line-height:0.0;
6589 }
6590 .GJWin{
6591     z-index:10000;
6592     display:inline;
6593     position:relative;
6594     flex-wrap: wrap;
6595     top:0; left:0px;
6596     width:285px !important; height:205px !important;
6597     border:1px solid #eea; border-radius:2px;
6598     margin:0px; padding:0px;
6599     font-size:8pt;
6600     line-height:0.0;
6601     color:#fff; background-color:rgba(0,0,64,0.1) !important;
6602 }
6603 .GJTab{
6604     display:inline;
6605     position:relative;
6606     top:0px; left:0px;
6607     margin:0px; padding:2px;
6608     border:0px solid #000; border-radius:2px;
6609     width:90px; height:20px;
6610     font-family:Georgia;
6611     font-size:9pt;
6612     line-height:1.0;
6613     white-space:nowrap;
6614     color:#fff; background-color:rgba(0,0,64,0.7);
6615     text-align:center;
6616     vertical-align:middle;
6617 }
6618 .GJStat:focus{
6619     color:#df8 !important;
6620     background-color:rgba(32,32,32,1.0) !important;
6621     line-height:1.0;
6622 }
6623 .GJStat{
6624     display:inline;
6625     position:relative;
6626     top:0px; left:0px;
6627     margin:0px; padding:2px;
6628     border:0px solid #00f; border-radius:2px;
6629     width:166px; height:20px;
6630     font-family:monospace;
6631     font-size:9pt;
6632     line-height:1.0;
6633     color:#fff; background-color:rgba(0,0,64,0.2);
6634     text-align:center;
6635     vertical-align:middle;
6636 }
6637 .GJIcon{
6638     display:inline;
6639     position:relative;
6640     top:0px; left:1px;
6641     border:2px solid #44a;
6642     margin:0px; padding:1px;
6643     width:13.2; height:13.2px;
6644     border-radius:2px;
6645     font-family:Georgia;
6646     font-size:13.2px;
6647     line-height:1.0;
6648     white-space:nowrap;
6649     color:#fff; background-color:rgba(32,32,160,0.8);
6650     text-align:center;
6651     vertical-align:middle;
6652     text-shadow:0px 0px;
6653 }
6654 .GJText:focus{
6655     color:#fff !important;
6656     background-color:rgba(32,32,160,0.8) !important;
6657     line-height:1.0;
6658 }
6659 .GJText{
6660     display:inline;
6661     position:relative;
6662     top:0px; left:0px;
6663     border:0px solid #000; margin:0px; padding:0px;
6664     width:280px; height:160px;
6665     border:0px;
6666     font-family:Courier New,monospace !important;
6667     font-size:8pt;
6668     line-height:1.0;
6669     white-space:pre;
6670     color:#fff; xbackground-color:rgba(0,0,64,0.5);
6671     background-color:rgba(32,32,128,0.8) !important;
6672 }
6673 .GJMode{
6674     display:inline;
6675     position:relative;
6676     top:0px; left:0px;
6677     border:0px solid #000; border-radius:0px;
6678     margin:0px; padding:0px;
6679     width:280px; height:20px;
6680     font-size:9pt;
6681     line-height:1.0;
6682     white-space:nowrap;
6683     color:#fff; background-color:rgba(0,0,64,0.7);
6684     text-align:left;
6685     vertical-align:middle;
6686 }
6687 </style>
6688
6689 <script id="gsh-script">
6690     // 2020-0909 added, permanent local storage
6691     // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
6692     var MyHistory = ""
6693     Permanent = localStorage;
6694     MyHistory = Permanent.getItem('MyHistory')
6695     if( MyHistory == null ){ MyHistory = "" }
6696     d = new Date()

```

```

6697 MyHistory = d.getTime()/1000+ " +document.URL+"\n" + MyHistory
6698 Permanent.setItem('MyHistory',MyHistory)
6699 //Permanent.setItem('MyWindow',window)
6700
6701 var GJLog_Win = null
6702 var GJLog_Tab = null
6703 var GJLog_Stat = null
6704 var GJLog_Text = null
6705 var GJWin_Mode = null
6706 var FProductInterval = 0
6707
6708 var GJ_FactoryID = -1
6709 var GJFactory = null
6710 if( e = document.getElementById('GJFactory_0') ){
6711     GJFactory_l.height = 0
6712     GJFactory = e
6713     e.setAttribute('class','GJFactory')
6714     var GJ_FactoryID = 0
6715 }else{
6716     GJFactory = GJFactory_1
6717     var GJ_FactoryID = 1
6718 }
6719
6720 function GJFactory_Destroy(){
6721     gjf = GJFactory
6722     //gjf = document.getElementById('GJFactory')
6723     //alert('gjf='+gjf)
6724     if( gjf != null ){
6725         if( gjf.childNodes != null ){
6726             for( i = 0; i < gjf.childNodes.length; i++ ){
6727                 gjf.removeChild(gjf.childNodes[i])
6728             }
6729         }
6730         gjf.innerHTML = ''
6731         gjf.style.width = 0
6732         gjf.style.height = 0
6733         gjf.removeAttribute('style')
6734         GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
6735         window.clearInterval(FProductInterval)
6736         return '-- Destroy: work product destroyed'
6737     }else{
6738         return '-- Destroy: work product not exist'
6739     }
6740 }
6741
6742 var TransMode = false
6743 var onKeyControl = false
6744 var OnKeyShift = false
6745 var OnKeyAlt = false
6746 var OnKeyJ = false
6747 var OnKeyK = false
6748 var OnKeyL = false
6749
6750 function GJWin_OnKeyUp(ev){
6751     keycode = ev.code;
6752     if( keycode == 'ShiftLeft' ){
6753         OnKeyShift = false
6754     }else
6755     if( keycode == 'ControlLeft' ){
6756         onKeyControl = false
6757     }else
6758     if( keycode == 'AltLeft' ){
6759         OnKeyAlt = false
6760     }else
6761     if( keycode == 'KeyJ' ){ OnKeyJ = false }else
6762     if( keycode == 'KeyK' ){ OnKeyK = false }else
6763     if( keycode == 'KeyL' ){ OnKeyL = false }else
6764     {
6765     }
6766     ev.preventDefault()
6767 }
6768 function and(a,b){ if(a){ if(b){ return true; } return false; } }
6769 function GJWin_OnKeyDown(ev){
6770     keycode = ev.code;
6771     mode = ''
6772     key = ''
6773     if( keycode == 'ControlLeft' ){
6774         onKeyControl = true
6775         ev.preventDefault()
6776         return;
6777     }else
6778     if( keycode == 'ShiftLeft' ){
6779         OnKeyShift = true
6780         ev.preventDefault()
6781         return;
6782     }else
6783     if( keycode == 'AltLeft' ){
6784         ev.preventDefault()
6785         OnKeyAlt = true
6786         return;
6787     }else
6788     if( keycode == 'Backquote' ){
6789         TransMode = !TransMode
6790         ev.preventDefault()
6791     }else
6792     if( and(keycode == 'Space', OnKeyShift) ){
6793         TransMode = !TransMode
6794         ev.preventDefault()
6795     }else
6796     if( keycode == 'ShiftRight' ){
6797         TransMode = !TransMode
6798     }else
6799     if( keycode == 'Escape' ){
6800         TransMode = true
6801         ev.preventDefault()
6802     }else
6803     if( keycode == 'Enter' ){
6804         TransMode = false
6805         //ev.preventDefault()
6806     }
6807     if( keycode == 'KeyJ' ){ OnKeyJ = true }else
6808     if( keycode == 'KeyK' ){ OnKeyK = true }else
6809     if( keycode == 'KeyL' ){ OnKeyL = true }else
6810     {
6811     }
6812
6813     if( ev.altKey ){ key += 'Alt+' }
6814     if( onKeyControl ){ key += 'Ctrl+' }
6815     if( OnKeyShift ){ key += 'Shift+' }
6816     if( and(keycode != 'KeyJ', OnKeyJ) ){ key += 'J+' }
6817     if( and(keycode != 'KeyK', OnKeyK) ){ key += 'K+' }
6818     if( and(keycode != 'KeyL', OnKeyL) ){ key += 'L+' }
6819     key += keycode
6820 }

```

```

6821     if( TransMode ){
6822         //mode = "\343\201\202r"
6823         mode = "[ðr]"
6824     }else{
6825         mode = '[-]'
6826     }
6827     ///// /gmode.innerHTML = "[---]"
6828     GJWin_Mode.innerHTML = mode + ' ' + key
6829     //alert('Key:'+keycode)
6830     ev.stopPropagation()
6831     //ev.preventDefault()
6832 }
6833 function GJWin_OnScroll(ev){
6834     x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
6835     y = DragStatty = gsh.getBoundingClientRect().top.toFixed(0)
6836     GJLog_append('OnScroll: x='+x+',y='+y)
6837 }
6838 document.addEventListener('scroll',GJWin_OnScroll)
6839 function GJWin_OnResize(ev){
6840     w = window.innerWidth
6841     h = window.innerHeight
6842     GJLog_append('OnResize: w='+w+',h='+h)
6843 }
6844 window.addEventListener('resize',GJWin_OnResize)
6845
6846 var DragStartX = 0
6847 var DragStartY = 0
6848 function GJWin_DragStart(ev){
6849     // maybe this is the grabbing position
6850     this.style.position = 'fixed'
6851     x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
6852     y = DragStatty = this.getBoundingClientRect().top.toFixed(0)
6853     GJLog_Stat.value = 'DragStart: x='+x+',y='+y
6854 }
6855 function GJWin_Drag(ev){
6856     x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
6857     this.style.left = x - DragStartX
6858     this.style.top = y - DragStartY
6859     this.style.zIndex = '30000'
6860     this.style.position = 'fixed'
6861     x = this.getBoundingClientRect().left.toFixed(0)
6862     y = this.getBoundingClientRect().top.toFixed(0)
6863     GJLog_Stat.value = 'x='+x+',y='+y
6864     ev.preventDefault()
6865     ev.stopPropagation()
6866 }
6867 function GJWin_DragEnd(ev){
6868     x = ev.clientX; y = ev.clientY
6869     //x = ev.pageX; y = ev.pageY
6870     this.style.left = x - DragStartX
6871     this.style.top = y - DragStartY
6872     this.style.zIndex = '30000'
6873     this.style.position = 'fixed'
6874     if( true ){
6875         console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y='+y
6876             +' parent='+this.parentNode.id)
6877     }
6878     x = this.getBoundingClientRect().left.toFixed(0)
6879     y = this.getBoundingClientRect().top.toFixed(0)
6880     GJLog_Stat.value = 'x='+x+',y='+y
6881     ev.preventDefault()
6882     ev.stopPropagation()
6883 }
6884 function GJWin_DragIgnore(ev){
6885     ev.preventDefault()
6886     ev.stopPropagation()
6887 }
6888 // 2020-09-15 let every object have console view!
6889 var GJ_ConsoleID = 0
6890 var PrevReport = new Date()
6891 function GJLog_StatUpdate(){
6892     txa = GJLog_Stat;
6893     if( txa == null ){
6894         return;
6895     }
6896     tmLap0 = new Date();
6897     p = txa.parentNode;
6898     pw = txa.getBoundingClientRect().width;
6899     ph = txa.getBoundingClientRect().height;
6900     //txa.value += '#'+p.id+' pw='+pw+', ph='+ph+'\n';
6901     tx1 = '#'+p.id+' pw='+pw+', ph='+ph+'\n';
6902
6903     w = txa.getBoundingClientRect().width;
6904     h = txa.getBoundingClientRect().height;
6905     //txa.value += 'w='+w+', h='+h+'\n';
6906     tx1 += 'w='+w+', h='+h+'\n';
6907
6908     //txa.value += '\n';
6909     //txa.value += DateShort() + '\n';
6910     tx1 += '\n';
6911     tx1 += DateShort() + '\n';
6912     tmLap1 = new Date();
6913
6914     txa.value += tx1;
6915     tmLap2 = new Date();
6916
6917     // vertical centering of the last line
6918     sHeight = txa.scrollHeight - 30; // depends on the font-size
6919     tmLap3 = new Date();
6920
6921     txa.scrollTop = sHeight; // depends on the font-size
6922     tmLap4 = new Date();
6923
6924     now = tmLap0.getTime();
6925     if( PrevReport == 0 || 10000 <= now-PrevReport ){
6926         PrevReport = now;
6927         console.log('StatBarUpdate:
6928             + 'len=' + txa.value.length + ' byte, '
6929             + 'times' + (tmLap4 - tmLap0) + 'ms {
6930             + 'tadd=' + (tmLap2 - tmLap1) + ', '
6931             + 'hcal=' + (tmLap3 - tmLap2) + ', '
6932             + 'sctl=' + (tmLap4 - tmLap3) + '}'
6933         );
6934     }
6935 }
6936 GJWin_StatUpdate = GJLog_StatUpdate;
6937 function GJ_showTime1(wid){
6938     //e = document.getElementById(wid);
6939     //console.log(wid.id+'.value.length='+wid.value.length)
6940     if( e != null ){
6941         //e.value = DateShort();
6942     }else{
6943         // should remove the Listener
6944     }

```



```

6945 }
6946 function GJWin_OnResizeTextarea(ev){
6947     this.value += 'resized:' + '\n'
6948 }
6949 function GJ_NewConsole(wname){
6950     wid = wname + '_' + GJ_ConsoleID
6951     GJ_ConsoleID += 1
6952
6953     GJFactory.style.setProperty('width',360+'px'); //GJFsize
6954     GJFactory.style.setProperty('height',320+'px')
6955     e = GJFactory;
6956     console.log('GJFa #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
6957
6958     if( GJFactory.innerHTML == "" ){
6959         GJFactory.innerHTML = '<' + 'H3>GJ_Factory_' + GJ_FactoryID + '<' + '/'H3><' + 'hr><' + '\n'
6960     }else{
6961         GJFactory.innerHTML += '<' + 'hr><' + '\n'
6962     }
6963
6964     gjwin = GJLog_Win = document.createElement('span')
6965     gjwin.id = wid
6966     gjwin.setAttribute('class','GJWin')
6967     gjwin.setAttribute('draggable','true')
6968     gjwin.addEventListener('dragstart',GJWin_DragStart)
6969     gjwin.addEventListener('drag',GJWin_Drag)
6970     gjwin.addEventListener('dragend',GJWin_Drag)
6971     gjwin.addEventListener('dragover',GJWin_DragIgnore)
6972     gjwin.addEventListener('dragenter',GJWin_DragIgnore)
6973     gjwin.addEventListener('dragleave',GJWin_DragIgnore)
6974     gjwin.addEventListener('dragexit',GJWin_DragIgnore)
6975     gjwin.addEventListener('drop',GJWin_DragIgnore)
6976     gjwin.addEventListener('keydown',GJWin_OnKeyDown)
6977
6978     gjtab = GJLog_Tab = document.createElement('textarea')
6979     gjtab.addEventListener('keydown',GJWin_OnKeyDown)
6980     gjtab.style.readonly = true
6981     gjtab.contentEditable = false
6982     gjtab.value = wid
6983     gjtab.id = wid + '_Tab'
6984     gjtab.setAttribute('class','GJTab')
6985     gjtab.setAttribute('spellcheck','false')
6986     gjwin.appendChild(gjtab)
6987
6988     gjstat = GJLog_Stat = document.createElement('textarea')
6989     gjstat.addEventListener('keydown',GJWin_OnKeyDown)
6990     gjstat.id = wid + '_Stat'
6991     gjstat.value = DateShort()
6992     gjstat.setAttribute('class','GJStat')
6993     gjstat.setAttribute('spellcheck','false')
6994     gjwin.appendChild(gjstat)
6995
6996     gjicon = document.createElement('span')
6997     gjicon.addEventListener('keydown',GJWin_OnKeyDown)
6998     gjicon.id = wid + '_Icon'
6999     gjicon.innerHTML = '<G;font color="#f44">J</font>'
7000     gjicon.setAttribute('class','GJIcon')
7001     gjicon.setAttribute('spellcheck','false')
7002     gjwin.appendChild(gjicon)
7003
7004     gjtext = GJLog_Text = document.createElement('textarea')
7005     gjtext.addEventListener('keydown',GJWin_OnKeyDown)
7006     gjtext.addEventListener('keyup',GJWin_OnKeyUp)
7007     gjtext.addEventListener('resize',GJWin_OnResizeTextarea)
7008     gjtext.id = wid + '_Text'
7009     gjtext.setAttribute('class','GJText')
7010     gjtext.setAttribute('spellcheck','false')
7011     gjwin.appendChild(gjtext)
7012
7013
7014     // user's mode as of IME
7015     gjmode = GJWin_Mode = document.createElement('textarea')
7016     gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7017     gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7018     gjmode.id = wid + '_Mode'
7019     gjmode.setAttribute('class','GJMode')
7020     gjmode.setAttribute('spellcheck','false')
7021     gjmode.innerHTML = '[--]'
7022     gjwin.appendChild(gjmode)
7023
7024     gjwin.zIndex = 30000
7025     GJFactory.appendChild(gjwin)
7026
7027     gjtab.scrollTop = 0
7028     gjstat.scrollTop = 0
7029
7030     //x = gjwin.getBoundingClientRect().left.toFixed(0)
7031     //y = gjwin.getBoundingClientRect().top.toFixed(0)
7032     //gjwin.style.position = 'static'
7033     //gjwin.style.left = 0
7034     //gjwin.style.top = 0
7035
7036     //update = '{'+wid+'}.value=DateShort()}',
7037     update = '{GJ_showTime1+' + wid + '}',
7038     // 2020-09-19 this causes memory leaks
7039     //FProductInterval = window.setInterval(update,200)
7040     //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
7041     //FProductInterval = window.setInterval(GJ_showTime1,200,wid);
7042     FProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
7043     return update
7044 }
7045 function xxxGJF_StripeClass(){
7046     GJLog_Win.style.removeProperty('width')
7047     GJLog_Tab.style.removeProperty('width')
7048     GJLog_Stat.style.removeProperty('width')
7049     GJLog_Text.style.removeProperty('width')
7050     return "Stripped classes"
7051 }
7052 function isElem(id){
7053     return document.getElementById(id) != null
7054 }
7055 function GJLog_append(...args){
7056     txt = GJLog_Text;
7057     if( txt == null ){
7058         return; // maybe GJLog element is removed
7059     }
7060     logs = args.join(' ')
7061     txt.value += logs + '\n'
7062     txt.scrollTop = txt.scrollHeight
7063     //GJLog_Stat.value = DateShort()
7064 }
7065 //window.addEventListener('time',GJLog_StatUpdate)
7066 function test_GJ_Console(){
7067     window.setInterval(GJLog_StatUpdate,1000);
7068     GJ_NewConsole('GJ_Console')

```

```

7069     e = GJFactory;
7070     console.log('GJF0 #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
7071     e.style.width = 360; //GJFsize
7072     e.style.height = 320;
7073     console.log('GJF0 #' + e.id + ' to w=' + e.style.width + ', h=' + e.style.height)
7074 }
7075 /// test_GJ_Console();
7076
7077 var StopConsoleLog = true
7078 // 2020-09-15 added,
7079 // log should be saved to permanent memory
7080 // const px = new Proxy(console.log, { alert() })
7081 __console_log = console.log
7082 __console_info = console.info
7083 __console_warn = console.warn
7084 __console_error = console.error
7085 __console_exception = console.exception
7086 // should pop callstack info.
7087 console.exception = function(...args){
7088     __console_exception(...args)
7089     alert('-- got console.exception("' + args + '"')
7090 }
7091 console.error = function(...args){
7092     __console_error(...args)
7093     alert('-- got console.error("' + args + '"')
7094 }
7095 console.warn = function(...args){
7096     __console_warn(...args)
7097     alert('-- got console.warn("' + args + '"')
7098 }
7099 console.info = function(...args){
7100     alert('-- got console.info("' + args + '"')
7101     __console_info(...args)
7102 }
7103 console.log = function(...args){
7104     __console_log(...args)
7105     if( StopConsoleLog ){
7106         return;
7107     }
7108     if( 0 <= args[0].indexOf('!') ){
7109         //alert('-- got console.log("' + args + '"')
7110     }
7111     GJLog_append(...args)
7112 }
7113
7114 //document.getElementById('GshFaviconURL').href = GShellFavicon
7115 document.getElementById('GshFaviconURL').href = GShellInsideIcon
7116 //document.getElementById('GshFaviconURL').href = ITSmoreQR
7117 //document.getElementById('GshFaviconURL').href = GShellLogo
7118
7119 // id of GShell HTML elements
7120 var E_BANNER = "GshBanner" // banner element in HTML
7121 var E_FOOTER = "GshFooter" // footer element in HTML
7122 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7123 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7124 var E_TODO = "gsh-todo" // TODO of GShell
7125 var E_DICT = "gsh-dict" // Dictionary of GShell
7126
7127 function bannerElem(){ return document.getElementById(E_BANNER); }
7128 function bannerStyleFunc(){ return bannerElem().style; }
7129 var bannerStyle = bannerStyleFunc()
7130 bannerStyle.backgroundImage = "url("+GShellLogo+")";
7131 //bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
7132 //bannerStyle.backgroundImage = "url("+GShellFavicon+")";
7133 GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7134
7135 function footerElem(){ return document.getElementById(E_FOOTER); }
7136 function footerStyle(){ return footerElem().style; }
7137 //footerElem().style.backgroundImage="url("+ITSmoreQR+")";
7138 //footerStyle().backgroundImage = "url("+ITSmoreQR+")";
7139
7140 function html_fold(e){
7141     if( e.innerHTML == "Fold" ){
7142         e.innerHTML = "Unfold"
7143         document.getElementById('gsh-menu-exit').innerHTML=""
7144         document.getElementById('GshStatement').open=false
7145         GshFeatures.open = false
7146         document.getElementById('html-src').open=false
7147         document.getElementById(E_GINDEX).open=false
7148         document.getElementById(E_GOCODE).open=false
7149         document.getElementById(E_TODO).open=false
7150         document.getElementById('References').open=false
7151     }else{
7152         e.innerHTML = "Fold"
7153         document.getElementById('GshStatement').open=true
7154         GshFeatures.open = true
7155         document.getElementById(E_GINDEX).open=true
7156         document.getElementById(E_GOCODE).open=true
7157         document.getElementById(E_TODO).open=true
7158         document.getElementById('References').open=true
7159     }
7160 }
7161 function html_pure(e){
7162     if( e.innerHTML == "Pure" ){
7163         document.getElementById('gsh').style.display=true
7164         //document.style.display = false
7165         e.innerHTML = "Unpure"
7166     }else{
7167         document.getElementById('gsh').style.display=false
7168         //document.style.display = true
7169         e.innerHTML = "Pure"
7170     }
7171 }
7172
7173 var bannerIsStopping = false
7174 //NOTE: .com/JSREF/prop_style_backgroundposition.asp
7175 function shiftBG(){
7176     bannerIsStopping = !bannerIsStopping
7177     bannerStyle.backgroundPosition = "0 0";
7178 }
7179 // status should be inherited on Window Fork(), so use the status in DOM
7180 function html_stop(e,toggle){
7181     if( toggle ){
7182         if( e.innerHTML == "Stop" ){
7183             bannerIsStopping = true
7184             e.innerHTML = "Start"
7185         }else{
7186             bannerIsStopping = false
7187             e.innerHTML = "Stop"
7188         }
7189     }else{
7190         // update JavaScript variable from DOM status
7191         if( e.innerHTML == "Stop" ){ // shown if it's running
7192             bannerIsStopping = false

```

```

7193     }else{
7194         bannerIsStopping = true
7195     }
7196 }
7197 }
7198 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
7199 //html_stop(bannerElem(),false) // onInit.
7200
7201 //https://www.w3schools.com/jsref/met_win_setinterval.asp
7202 function shiftBanner(){
7203     var now = new Date().getTime();
7204     //console.log("now="+(now%10))
7205     if( !bannerIsStopping ){
7206         bannerStyle.backgroundPosition = ((now/10)%100000)+" 0";
7207     }
7208 }
7209 window.setInterval(shiftBanner,10); // onInit.
7210
7211 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
7212 // from embedded html to standalone page
7213 var MyChildren = 0
7214 function html_fork(){
7215     GJFactory_Destroy()
7216     MyChildren += 1
7217     WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
7218     newwin = window.open("",WinId,"");
7219     src = document.getElementById("gsh");
7220     srchtml = src.outerHTML
7221     newwin.document.write("/*<"+html>\n");
7222     newwin.document.write(srchtml);
7223     newwin.document.write("<"+html>\n");
7224     newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
7225     newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
7226     newwin.document.close();
7227     newwin.focus();
7228 }
7229 function html_close(){
7230     window.close()
7231 }
7232 function win_jump(win){
7233     //win = window.top;
7234     win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
7235     if( win == null ){
7236         console.log("jump to window.opener("+win+") (Error)\n")
7237     }else{
7238         console.log("jump to window.opener("+win+")\n")
7239         win.focus();
7240     }
7241 }
7242
7243 // 0.2.9 2020-0902 created chekcosum of HTML
7244 CRC32UNIX = 0x04C11DB7 // Unix cksum
7245 function byteCRC32add(bigcrc,octstr,octlen){
7246     var crc = new Int32Array(1)
7247     crc[0] = bigcrc
7248
7249     let oi = 0
7250     for( ; oi < octlen; oi++){
7251         var oct = new Int8Array(1)
7252         oct[0] = octstr[oi]
7253         for( bi = 0; bi < 8; bi++){
7254             //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+"."+bi+"]\n")
7255             ovf1 = crc[0] < 0 ? 1 : 0
7256             ovf2 = oct[0] < 0 ? 1 : 0
7257             ovf = ovf1 ^ ovf2
7258             oct[0] <<= 1
7259             crc[0] <<= 1
7260             if( ovf ){ crc[0] ^= CRC32UNIX }
7261         }
7262     }
7263     //console.log("--CRC32 byteAdd return crc="+crc[0]+","+oi+"/"+"octlen+"\n")
7264     return crc[0];
7265 }
7266 function strCRC32add(bigcrc,stri,strlen){
7267     var crc = new Uint32Array(1)
7268     crc[0] = bigcrc
7269     var code = new Uint8Array(strlen);
7270     for( i = 0; i < strlen; i++){
7271         code[i] = stri.charCodeAtAt(i) // not charAt() !!!!
7272         //console.log("=== "+code[i].toString(16)+" <<= "+stri[i]+"")
7273     }
7274     crc[0] = byteCRC32add(crc,code,strlen)
7275     //console.log("--CRC32 strAdd return crc="+crc[0]+"")
7276     return crc[0]
7277 }
7278 function byteCRC32end(bigcrc,len){
7279     var crc = new Uint32Array(1)
7280     crc[0] = bigcrc
7281     var slen = new Uint8Array(4)
7282     let li = 0
7283     for( ; li < 4; ){
7284         slen[li] = len
7285         li += 1
7286         len >>= 8
7287         if( len == 0 ){
7288             break
7289         }
7290     }
7291     crc[0] = byteCRC32add(crc[0],slen,li)
7292     crc[0] ^= 0xFFFFFFFF
7293     return crc[0]
7294 }
7295 function strCRC32(stri,len){
7296     var crc = new Uint32Array(1)
7297     crc[0] = 0
7298     crc[0] = strCRC32add(0,stri,len)
7299     crc[0] = byteCRC32end(crc[0],len)
7300     //console.log("--CRC32 "+crc[0]+" "+len+"\n")
7301     return crc[0]
7302 }
7303
7304 DestroyGJLink = null; // to be replaced
7305 DestroyFooter = null; // to be defined
7306 DestroyEventSharingCodeview = function dummy(){}
7307 DestroyWirtualDesktop = function(){}
7308 DestroyNaviButtons = function(){}
7309
7310 function getSourceText(){
7311     if( DestroyFooter != null ) DestroyFooter();
7312     version = document.getElementById('GshVersion').innerHTML
7313     sfavico = document.getElementById('GshFaviconURL').href;
7314     sbanner = document.getElementById('GshBanner').style.backgroundImage;
7315     spositi = document.getElementById('GshBanner').style.backgroundPosition;
7316

```

```

7317 if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
7318 if( DestroyGJLink != null ) DestroyGJLink();
7319 DestroyEventSharingCodeview();
7320 Destroy_WirtualDesktop();
7321 Destroy_IndexBar();
7322 DestroyNaviButtons();
7323
7324 // these should be removed by CSS selector or class, after sevaed to non-printed attribute
7325 GshBanner.removeAttribute('style');
7326 document.getElementById('GshMenuSign').removeAttribute("style");
7327 styleGMenu = GMenu.getAttribute("style")
7328 GMenu.removeAttribute("style");
7329 styleGStat = GStat.getAttribute("style")
7330 GStat.removeAttribute("style");
7331 styleGTop = GTop.getAttribute("style")
7332 GTop.removeAttribute("style");
7333 styleGshGrid = GshGrid.getAttribute("style")
7334 GshGrid.removeAttribute("style");
7335 //styleGPos = GPos.getAttribute("style");
7336 //GPos.removeAttribute("style");
7337 //GPos.innerHTML = "";
7338 //styleGLog = GLog.getAttribute("style");
7339 //GLog.removeAttribute("style");
7340 //GLog.innerHTML = "";
7341 styleGShellPlane = GShellPlane.getAttribute("style")
7342 GShellPlane.removeAttribute("style")
7343 styleRawTextViewer = RawTextViewer.getAttribute("style")
7344 RawTextViewer.removeAttribute("style")
7345 styleRawTextViewerClose = RawTextViewerClose.getAttribute("style")
7346 RawTextViewerClose.removeAttribute("style")
7347
7348 GshFaviconURL.href = "";
7349 ConfigIcon.src = "";
7350
7351 //it seems that interHTML and outerHTML generate style="" for these (??)
7352 //GshBanner.removeAttribute('style');
7353 //GshFooter.removeAttribute('style');
7354 //GshMenuSign.removeAttribute('style');
7355 GshBanner.style="";
7356 GshMenuSign.style=""
7357
7358 textarea = document.createElement("textarea")
7359 srchtml = document.getElementById("gsh").outerHTML;
7360 //textarea = document.createElement("textarea")
7361 // 2020-0910 ?? ... this causes inserting style="" to Banner and Footer,
7362 // with Chromium?/ after reloading from file:///
7363 textarea.innerHTML = srchtml
7364 // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7365 var rawtext = textarea.value
7366 //textarea.destroy()
7367 //rawtext = gsh.textContent // this removes #include <FILENAME> too
7368 var orgtext = ""
7369 + /*<+>html>\n" // lost preamble text
7370 + rawtext
7371 + "<+>/html>\n" // lost trail text
7372 ;
7373
7374 tlen = orgtext.length
7375 //console.log("getSourceText: length="+tlen+"\n")
7376 document.getElementById('GshFaviconURL').href = sfavico;
7377
7378 document.getElementById('GshBanner').style.backgroundImage = sbanner;
7379 document.getElementById('GshBanner').style.backgroundPosition = spositi;
7380
7381 GStat.setAttribute("style",styleGStat)
7382 GMenu.setAttribute("style",styleGMenu)
7383 GTop.setAttribute("style",styleGTop)
7384 //GLog.setAttribute("style",styleGLog)
7385 //GPos.setAttribute("style",styleGPos)
7386 GshGrid.setAttribute("style",styleGshGrid)
7387 GShellPlane.setAttribute("style",styleGShellPlane)
7388 RawTextViewer.setAttribute("style",styleRawTextViewer)
7389 RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
7390 canontext = orgtext.replace(' style=""','')
7391 // open="" too
7392 return canontext
7393 }
7394 function getDigest(){
7395 var text = ""
7396 text = getSourceText()
7397 var digest = ""
7398 tlen = text.length
7399 digest = strCRC32(text,tlen) + " " + tlen
7400 return { text, digest }
7401 }
7402 function html_digest(){
7403 version = document.getElementById('GshVersion').innerHTML
7404 let {text, digest} = getDigest()
7405 alert("cksum: " + digest + " " + version)
7406 }
7407 function charsin(stri,char){
7408 ln = 0;
7409 for( i = 0; i < stri.length; i++){
7410 if( stri.charCodeAt(i) == char.charCodeAt(0) )
7411 ln++;
7412 }
7413 return ln;
7414 }
7415
7416 //class digestElement extends HTMLElement { }
7417 //< script>customElements.define('digest',digestElement)< /script>
7418 function showDigest(e){
7419 result = 'version=' + GshVersion.innerHTML + '\n'
7420 result += 'lines=' + e.dataset.lines + '\n'
7421 + 'length=' + e.dataset.length + '\n'
7422 + 'crc32u=' + e.dataset.crc32u + '\n'
7423 + 'time=' + e.dataset.time + '\n';
7424
7425 alert(result)
7426 }
7427
7428 function html_sign(e){
7429 if( RawTextViewer.style.zIndex == 1000 ){
7430 hideRawTextViewer()
7431 return
7432 }
7433 DestroyIndexBar();
7434 DestroyNaviButtons();
7435 DestroyEventSharingCodeview();
7436 Destroy_WirtualDesktop();
7437 GJFactory_Destroy()
7438 if( DestroyGJLink != null ) DestroyGJLink();
7439 //gsh_digest_.innerHTML = "";
7440 text = getSourceText() // the original text

```

```

7441 tlen = text.length
7442 digest = strCRC32(text,tlen)
7443 //gsh_digest_.innerHTML = digest + " " + tlen
7444 //text = getSourceText() // the text with its digest
7445 Lines = charsIn(text, '\n')
7446
7447 name = "gsh"
7448 sid = name + "--digest"
7449 d = new Date()
7450 signedAt = d.getTime()
7451
7452 sign = '/'+'*'+span'\n'
7453 + ' id="' + sid + "'\n'
7454 + ' class=" digest_"\n'
7455 + ' data-target-id="'+name+'"\n'
7456 + ' data-crc32u="' + digest + "'\n'
7457 + ' data-length="' + tlen + "'\n'
7458 + ' data-lines="' + Lines + "'\n'
7459 + ' data-time="' + signedAt + "'\n'
7460 + '>' + '/span>\n'+'/\n'
7461
7462 text = sign + text
7463
7464 txhtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td>'
7465 + '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">'
7466 for( i = 1; i <= Lines; i++){
7467     txhtml += i.toString() + '\n'
7468 }
7469 txhtml += ""
7470 + '<' + '/textarea>'
7471 + '<' + '/td><' + 'td>'
7472 + '<' + 'textarea cols=150 rows=' + Lines + ' spellcheck="false"'
7473 + ' class="LineNumber">'
7474 + text + '<' + '/textarea>'
7475 + '<' + '/td><' + '/tr><' + '/table>'
7476
7477 for( i = 1; i <= 30; i++){
7478     txhtml += '<br>\n'
7479 }
7480 RawTextViewer.innerHTML = txhtml
7481 RawTextViewer.spellcheck = false // (spellcheck above seems ineffective)
7482
7483 btn = e
7484 e.style.color = "rgba(128,128,255,0.9)";
7485 y = e.getBoundingClientRect().top.toFixed(0)
7486 //h = e.getBoundingClientRect().height.toFixed(0)
7487 RawTextViewer.style.top = Number(y) + 30
7488 RawTextViewer.style.left = 100;
7489 RawTextViewer.style.height = window.innerHeight - 20;
7490 //RawTextViewer.style.Opacity = 1.0;
7491 //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
7492 RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
7493 RawTextViewer.style.zIndex = 1000;
7494 RawTextViewer.style.display = true;
7495
7496 if( RawTextViewerClose.style == null ){
7497     RawTextViewerClose.style = "";
7498 }
7499 RawTextViewerClose.style.top = Number(y) + 10
7500 RawTextViewerClose.style.left = 100;
7501 RawTextViewerClose.style.zIndex = 1001;
7502
7503 ScrollToElement(CurElement,RawTextViewerClose)
7504 }
7505 function hideRawTextViewer(){
7506     RawTextViewer.style.left = 10000;
7507     RawTextViewer.style.zIndex = -100;
7508     RawTextViewer.style.Opacity = 0.0;
7509     RawTextViewer.style = null
7510     RawTextViewer.innerHTML = "";
7511
7512     GshMenuSign.style.color = "rgba(255,128,128,1.0)";
7513     RawTextViewerClose.style.top = 0;
7514     RawTextViewerClose.style = null
7515 }
7516
7517 // source code viewr
7518 function frame_close(){
7519     srcframe = document.getElementById("src-frame");
7520     srcframe.innerHTML = "";
7521     //srcframe.style.cols = 1;
7522     srcframe.style.rows = 1;
7523     srcframe.style.height = 0;
7524     srcframe.style.display = false;
7525     src = document.getElementById("SrcTextarea");
7526     src.innerHTML = ""
7527     //src.cols = 0
7528     src.rows = 0
7529     src.display = false
7530     //alert("--closed--")
7531 }
7532 //<!-- | <span onclick="html_view();">Source</span> -->
7533 //<!-- | <span onclick="frame_close();">SourceClose</span> -->
7534 //<!--| <span>Download</span> -->
7535 function frame_open(){
7536     DestroyIndexBar();
7537     DestroyNaviButtons();
7538     if( DestroyFooter != null ) DestroyFooter();
7539     document.getElementById('GshFaviconURL').href = "";
7540     oldsrc = document.getElementById("GENSRC");
7541     if( oldsrc != null ){
7542         //alert("--I--(erasing old text)")
7543         oldsrc.innerHTML = "";
7544         return
7545     }else{
7546         //alert("--I--(no old text)")
7547     }
7548     styleBanner = GshBanner.getAttribute("style")
7549     GshBanner.removeAttribute("style")
7550     if( document.getElementById('GJC_1') ){ GJC_1.remove() }
7551
7552     GshFaviconURL.href = "";
7553     ConfigIcon.src = "";
7554     GStat.removeAttribute('style')
7555     GshGrid.removeAttribute('style')
7556     GshMenuSign.removeAttribute('style')
7557     //GPos.removeAttribute('style')
7558     //GPos.innerHTML = "";
7559     //GLog.removeAttribute('style')
7560     //GLog.innerHTML = "";
7561     GMenu.removeAttribute('style')
7562     GTop.removeAttribute('style')
7563     GShellPlane.removeAttribute('style')
7564     RawTextViewer.removeAttribute('style')

```

```

7565 RawTextViewerClose.removeAttribute('style')
7566
7567 if( DestroyGJLink != null ) DestroyGJLink();
7568 GJFactory.Destroy();
7569 Destroy_WirtualDesktop();
7570 DestroyEventSharingCodeview();
7571
7572 src = document.getElementById("gsh");
7573 srchtml = src.outerHTML
7574 srcframe = document.getElementById("src-frame");
7575 srcframe.innerHTML = ""
7576 + "<+\"cite id=\"GENSRC\">\n"
7577 + "<+\"style>\n"
7578 + "#GENSRC textarea{tab-size:4;}\n"
7579 + "#GENSRC textarea{-o-tab-size:4;}\n"
7580 + "#GENSRC textarea{-moz-tab-size:4;}\n"
7581 + "#GENSRC textarea{spellcheck:false;}\n"
7582 + "</+\"style>\n"
7583 + "<+\"textarea id=\"SrcTextarea\" cols=100 rows=20 class=\"gsh-code\" spellcheck=\"false\">"
7584 + "/*<+\"html>\n" // lost preamble text
7585 + srchtml
7586 + "<+\"/html>\n" // lost trail text
7587 + "</+\"textarea>\n"
7588 + "</+\"cite><!-- GENSRC -->\n";
7589
7590 //srcframe.style.cols = 80;
7591 //srcframe.style.rows = 80;
7592
7593 GshBanner.setAttribute('style',styleBanner)
7594 }
7595 function fill_CSSView(){
7596 part = document.getElementById('GshStyleDef')
7597 view = document.getElementById('gsh-style-view')
7598 view.innerHTML = ""
7599 + "<+\"textarea cols=100 rows=20 class=\"gsh-code\">"
7600 + part.innerHTML
7601 + "<+\"/textarea>"
7602 }
7603 function fill_JavaScriptView(){
7604 jspart = document.getElementById('gsh-script')
7605 view = document.getElementById('gsh-script-view')
7606 view.innerHTML = ""
7607 + "<+\"textarea cols=100 rows=20 class=\"gsh-code\">"
7608 + jspart.innerHTML
7609 + "<+\"/textarea>"
7610 }
7611 function fill_DataView(){
7612 part = document.getElementById('gsh-data')
7613 view = document.getElementById('gsh-data-view')
7614 view.innerHTML = ""
7615 + "<+\"textarea cols=100 rows=20 class=\"gsh-code\">"
7616 + part.innerHTML
7617 + "<+\"/textarea>"
7618 }
7619 function jumpto_StyleView(){
7620 jsview = document.getElementById('html-src')
7621 jsview.open = true
7622 jsview = document.getElementById('gsh-style-frame')
7623 jsview.open = true
7624 fill_CSSView()
7625 }
7626 function jumpto_JavaScriptView(){
7627 jsview = document.getElementById('html-src')
7628 jsview.open = true
7629 jsview = document.getElementById('gsh-script-frame')
7630 jsview.open = true
7631 fill_JavaScriptView()
7632 }
7633 function jumpto_DataView(){
7634 jsview = document.getElementById('html-src')
7635 jsview.open = true
7636 jsview = document.getElementById('gsh-data-frame')
7637 jsview.open = true
7638 fill_DataView()
7639 }
7640 function jumpto_WholeView(){
7641 jsview = document.getElementById('html-src')
7642 jsview.open = true
7643 jsview = document.getElementById('gsh-whole-view')
7644 jsview.open = true
7645 frame_open()
7646 }
7647 function html_view(){
7648 html_stop();
7649
7650 banner = document.getElementById('GshBanner').style.backgroundImage;
7651 footer = document.getElementById('GshFooter').style.backgroundImage;
7652 document.getElementById('GshBanner').style.backgroundImage = "";
7653 document.getElementById('GshBanner').style.backgroundPosition = "";
7654 document.getElementById('GshFooter').style.backgroundImage = "";
7655
7656 //srcwin = window.open("", "CodeView2", "");
7657 srcwin = window.open("", "", "");
7658 srcwin.document.write("<span id=\"gsh\">\n");
7659
7660 src = document.getElementById("gsh");
7661 srcwin.document.write("<+\"style>\n");
7662 srcwin.document.write("textarea{tab-size:4;}\n");
7663 srcwin.document.write("textarea{-o-tab-size:4;}\n");
7664 srcwin.document.write("textarea{-moz-tab-size:4;}\n");
7665 srcwin.document.write("</style>\n");
7666 srcwin.document.write("<h2>\n");
7667 srcwin.document.write("<+\"span onclick=\"window.close();\">Close</span> | \n");
7668 //srcwin.document.write("<+\"span onclick=\"html_stop();\">Run</span>\n");
7669 srcwin.document.write("</h2>\n");
7670 srcwin.document.write("<+\"textarea id=\"gsh-src-src\" cols=100 rows=60>");
7671 srcwin.document.write("/*<+\"html>\n");
7672 srcwin.document.write("<+\"span id=\"gsh\">");
7673 srcwin.document.write(src.innerHTML);
7674 srcwin.document.write("<+\"/span><+\"/html>\n");
7675 srcwin.document.write("</+\"textarea>\n");
7676
7677 document.getElementById('GshBanner').style.backgroundImage = banner;
7678 document.getElementById('GshFooter').style.backgroundImage = footer
7679
7680 sty = document.getElementById('GshStyleDef');
7681 srcwin.document.write("<+\"style>\n");
7682 srcwin.document.write(sty.innerHTML);
7683 srcwin.document.write("<+\"/style>\n");
7684
7685 run = document.getElementById("gsh-script");
7686 srcwin.document.write("<+\"script>\n");
7687 srcwin.document.write(run.innerHTML);
7688 srcwin.document.write("<+\"/script>\n");

```

```

7689
7690 srcwin.document.write("<+</span><+</html>\n"); // gsh span
7691 srcwin.document.close();
7692 srcwin.focus();
7693 }
7694 GSH = document.getElementById("gsh")
7695
7696 //GSH.onclick = "alert('Ouch!')"
7697 //GSH.css = "{background-color:#eef;}"
7698 //GSH.style = "background-color:#eef;"
7699 //GSH.style.display = false;
7700 //alert('Ouch0!')
7701 //GSH.style.display = true;
7702
7703 // 2020-0904 created, tentative
7704 document.addEventListener('keydown',jgshCommand);
7705 //CurElement = GshStatement
7706 CurElement = GshMenu
7707 MemElement = GshMenu
7708
7709 function nextSib(e){
7710     n = e.nextSibling;
7711     for( i = 0; i < 100; i++ ){
7712         if( n == null ){
7713             break;
7714         }
7715         if( n.nodeName == "DETAILS" ){
7716             return n;
7717         }
7718         n = n.nextSibling;
7719     }
7720     return null;
7721 }
7722 function prevSib(e){
7723     n = e.previousSibling;
7724     for( i = 0; i < 100; i++ ){
7725         if( n == null ){
7726             break;
7727         }
7728         if( n.nodeName == "DETAILS" ){
7729             return n;
7730         }
7731         n = n.previousSibling;
7732     }
7733     return null;
7734 }
7735 function setColor(e,eName,eColor){
7736     if( e.hasChildNodes() ){
7737         s = e.childNodes;
7738         if( s != null ){
7739             for( ci = 0; ci < s.length; ci++ ){
7740                 if( s[ci].nodeName == eName ){
7741                     s[ci].style.color = eColor;
7742                     //s[ci].style.backgroundColor = eColor;
7743                     break;
7744                 }
7745             }
7746         }
7747     }
7748 }
7749
7750 // https://docs.microsoft.com/en-us/previous-versions//hh781509(v=vs.85)
7751 function showCurElementPosition(ev){
7752 // if( document.getElementById("GPos") == null ){
7753 //     return;
7754 // }
7755 // if( GPos == null ){
7756 //     return;
7757 // }
7758 e = CurElement
7759 y = e.getBoundingClientRect().top.toFixed(0)
7760 x = e.getBoundingClientRect().left.toFixed(0)
7761
7762 h = ev + " "
7763 h += 'y='+y+', ' + 'x='+x' -- "
7764 h += "w=" + window.innerWidth + ", h=" + window.innerHeight + " -- "
7765 //GPos.test = h
7766 //GPos.innerHTML = h
7767 // GPos.innerHTML = h
7768 }
7769
7770 function zero2(n){
7771     if( n < 10 ){
7772         return '0' + n;
7773     }else{
7774         return n;
7775     }
7776 }
7777 function DateHourMin(){
7778     d = new Date();
7779     //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes())
7780     return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
7781 }
7782 function DateShort0(d){
7783     return d.getFullYear()
7784         + '/' + zero2(d.getMonth())
7785         + '/' + zero2(d.getDate())
7786         + ' ' + zero2(d.getHours())
7787         + ':' + zero2(d.getMinutes())
7788         + ':' + zero2(d.getSeconds())
7789 }
7790 function DateShort(){
7791     return DateShort0(new Date());
7792 }
7793 function DateLong0(ms){
7794     d = new Date();
7795     d.setTime(ms);
7796     return DateShort0(d)
7797         + '.' + d.getMilliseconds()
7798         + '.' + d.getTimezoneOffset()/60
7799         + '.' + d.getTime() + '.' + d.getMilliseconds()
7800 }
7801 function DateLong(){
7802     return DateLong0(new Date());
7803 }
7804 function GShellMenu(e){
7805     //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
7806     showGShellPlane()
7807 }
7808 // placements of planes
7809 function GShellResizeX(ev){
7810     //if( document.getElementById("GMenu") != null ){
7811         GMenu.style.left = window.innerWidth - 100
7812         GMenu.style.top = window.innerHeight - 90 - 200

```

```

7813 //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)
7814
7815 //}
7816 GStat.style.width = window.innerWidth
7817 //if( document.getElementById("GPos") != null ){
7818 //GPos.style.width = window.innerWidth
7819 //GPos.style.top = window.innerHeight - 30; //GPos.style.height
7820 //}
7821 //if( document.getElementById("GLog") != null ){
7822 // GLog.style.width = window.innerWidth
7823 //GLog.innerHTML = ""
7824 //}
7825 //if( document.getElementById("GLog") != null ){
7826 //GLog.innerHTML = "Resize: w=" + window.innerWidth +
7827 //", h=" + window.innerHeight
7828 //}
7829 showCurElementPosition(ev)
7830 }
7831 function GShellResize(){
7832 GShellResizeX("[RESIZE]")
7833 }
7834 window.onresize = GShellResize
7835 var prevNode = null
7836 var LogMouseMoveOverElement = false;
7837 function GJSH_OnMouseMove(ev){
7838 if( LogMouseMoveOverElement == false ){
7839 return;
7840 }
7841 x = ev.clientX
7842 y = ev.clientY
7843 d = new Date()
7844 t = d.getTime() / 1000
7845 if( document.elementFromPoint(x,y) )
7846 e = document.elementFromPoint(x,y)
7847 if( e != null ){
7848 if( e != prevNode ){
7849 }else{
7850 console.log('Mo-'+t+'('+x+', '+y+') '
7851 '+e.nodeType+' '+e.tagName+'#'+e.id)
7852 prevNode = e
7853 }
7854 }else{
7855 console.log(t+'('+x+', '+y+') no element')
7856 }
7857 }else{
7858 console.log(t+'('+x+', '+y+') no elementFromPoint')
7859 }
7860 }
7861 window.addEventListener('mousemove',GJSH_OnMouseMove);
7862
7863 function GJSH_OnMouseMoveScreen(ev){
7864 x = ev.screenX
7865 y = ev.screenY
7866 d = new Date()
7867 t = d.getTime() / 1000
7868 console.log(t+'('+x+', '+y+') no elementFromPoint')
7869 }
7870 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
7871
7872 function ScrollToElement(oe,ne){
7873 ne.scrollIntoView()
7874 ny = ne.getBoundingClientRect().top.toFixed(0)
7875 nx = ne.getBoundingClientRect().left.toFixed(0)
7876 //GLog.innerHTML = "["+ny+", "+nx+"]"
7877 //window.scrollTo(0,0)
7878
7879 GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
7880 GshGrid.style.left = '250px';
7881 GshGrid.style.zIndex = 0
7882 if( false ){
7883 oy = oe.getBoundingClientRect().top.toFixed(0)
7884 ox = oe.getBoundingClientRect().left.toFixed(0)
7885 y = e.getBoundingClientRect().top.toFixed(0)
7886 x = e.getBoundingClientRect().left.toFixed(0)
7887 window.scrollTo(x,y)
7888 ny = e.getBoundingClientRect().top.toFixed(0)
7889 nx = e.getBoundingClientRect().left.toFixed(0)
7890 //GLog.innerHTML = "["+oy+", "+ox+"->["+y+", "+x+"->["+ny+", "+nx+"]"
7891 }
7892 }
7893 function showGShellPlane(){
7894 if( GShellPlane.style.zIndex == 0 ){
7895 GShellPlane.style.zIndex = 1000;
7896 GShellPlane.style.left = 30;
7897 GShellPlane.style.height = 320;
7898 GShellPlane.innerHTML = DateLong() + "<br> +
7899 "-- History --<br>" + MyHistory;
7900 }else{
7901 GShellPlane.style.zIndex = 0;
7902 GShellPlane.style.left = 0;
7903 GShellPlane.style.height = 50;
7904 GShellPlane.innerHTML = "";
7905 }
7906 }
7907 var SuppressGJShell = false
7908 function jgshCommand(kevent){
7909 if( SuppressGJShell ){
7910 return
7911 }
7912 key = kevent
7913 keycode = key.code
7914 //GStat.style.width = window.innerWidth
7915 GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
7916
7917 console.log("JSGsh-Key: "+keycode+"(^-^)//")
7918 if( keycode == "Slash" ){
7919 console.log('('+x+', '+y+') '
7920 e = document.elementFromPoint(x,y)
7921 console.log('('+x+', '+y+') '+e.nodeType+' '+e.tagName+'#'+e.id)
7922 }else
7923 if( keycode == "Digit0" ){ // fold side-bar
7924 // "Zero page"
7925 showGShellPlane();
7926 }else
7927 if( keycode == "Digit1" ){ // fold side-bar
7928 primary.style.width = "94%"
7929 secondary.style.width = "0%"
7930 secondary.style.opacity = 0
7931 GStat.innerHTML = "[Single Column View]"
7932 }else
7933 if( keycode == "Digit2" ){ // unfold side-bar
7934 primary.style.width = "58%"
7935 secondary.style.width = "36%"
7936 secondary.style.opacity = 1

```



```

7937     GStat.innerHTML = "[Double Column View]"
7938 }else
7939 if( keycode == "KeyU" ){ // fold/unfold all
7940     html_fold(GshMenuFold);
7941     location.href = "#"+CurElement.id;
7942 }else
7943 if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
7944     CurElement.open = !CurElement.open;
7945 }else
7946 if( keycode == "ArrowRight" ){ // unfold the element
7947     CurElement.open = true
7948 }else
7949 if( keycode == "ArrowLeft" ){ // unfold the element
7950     CurElement.open = false
7951 }else
7952 if( keycode == "KeyI" ){ // inspect the element
7953     e = CurElement
7954     //GLog.innerHTML +=
7955     GJLog_append("Current Element: " + e + "<br>"
7956         + "name="+e.nodeName + ", "
7957         + "id="+e.id + ", "
7958         + "children="+e.childNodes.length + ", "
7959         + "parent="+e.parentNode.id + "<br>"
7960         + "text="+e.textContent)
7961     GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
7962     return
7963 }else
7964 if( keycode == "KeyM" ){ // memory the position
7965     MemElement = CurElement
7966 }else
7967 if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
7968     e = nextSib(CurElement)
7969     if( e != null ){
7970         setColor(CurElement,"SUMMARY","#fff")
7971         setColor(e,"SUMMARY","#8f8") // should be complement ?
7972         oe = CurElement
7973         CurElement = e
7974         //location.href = "#"+e.id;
7975         ScrollToElement(oe,e)
7976     }
7977 }else
7978 if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
7979     oe = CurElement
7980     e = prevSib(CurElement)
7981     if( e != null ){
7982         setColor(CurElement,"SUMMARY","#fff")
7983         setColor(e,"SUMMARY","#8f8") // should be complement ?
7984         CurElement = e
7985         //location.href = "#"+e.id;
7986         ScrollToElement(oe,e)
7987     }else{
7988         e = document.getElementById("GshBanner")
7989         if( e != null ){
7990             setColor(CurElement,"SUMMARY","#fff")
7991             CurElement = e
7992             ScrollToElement(oe,e)
7993         }else{
7994             e = document.getElementById("primary")
7995             if( e != null ){
7996                 setColor(CurElement,"SUMMARY","#fff")
7997                 CurElement = e
7998                 ScrollToElement(oe,e)
7999             }
8000         }
8001     }
8002 }else
8003 if( keycode == "KeyR" ){
8004     location.reload()
8005 }else
8006 if( keycode == "KeyJ" ){
8007     GshGrid.style.top = '120px';
8008     GshGrid.innerHTML = '><{Down}';
8009 }else
8010 if( keycode == "KeyK" ){
8011     GshGrid.style.top = '0px';
8012     GshGrid.innerHTML = '^~^{Up}';
8013 }else
8014 if( keycode == "KeyH" ){
8015     GshGrid.style.left = '0px';
8016     GshGrid.innerHTML = "(_){Left}";
8017 }else
8018 if( keycode == "KeyL" ){
8019     //GLog.innerHTML +=
8020     GJLog_append(
8021         "screen="+screen.width+'px'+<br>'+
8022         "window="+window.innerWidth+'px'+<br>'+
8023     )
8024     GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
8025     GshGrid.innerHTML = '@_{Right}';
8026 }else
8027 if( keycode == "KeyS" ){
8028     html_stop(GshMenuStop,true)
8029 }else
8030 if( keycode == "KeyF" ){
8031     html_fork()
8032 }else
8033 if( keycode == "KeyC" ){
8034     window.close()
8035 }else
8036 if( keycode == "KeyD" ){
8037     html_digest()
8038 }else
8039 if( keycode == "KeyV" ){
8040     e = document.getElementById('gsh-digest')
8041     if( e != null ){
8042         showDigest(e)
8043     }
8044 }
8045 }
8046 showCurElementPosition("[+key.code+] --");
8047 //if( document.getElementById("GPos") != null ){
8048     //GPos.innerHTML += "[+key.code+] --"
8049 //}
8050 //GShellResizeX("[+key.code+] --");
8051 }
8052 GShellResizeX("[INIT]");
8053
8054 DisplaySize = '-- Display: '+ 'screen='+screen.width+'px, '+ 'window='+window.innerWidth+'px';
8055
8056 let {text, digest} = getDigest()
8057 //GLog.innerHTML +=
8058 GJLog_append(
8059     "-- Gshell: ' + GshVersion.innerHTML + '\n' +
8060     "-- Digest: ' + digest + '\n' +

```

```

8061     DisplaySize
8062     //+ "<br>" + "-- LastVisit:<br>" + MyHistory
8063 )
8064 GShellResizeX(null);
8065
8066 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8067 //Convert a string into an ArrayBuffer
8068 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8069 function str2ab(str) {
8070     const buf = new ArrayBuffer(str.length);
8071     const bufView = new Uint8Array(buf);
8072     for (let i = 0, strLen = str.length; i < strLen; i++) {
8073         bufView[i] = str.charCodeAt(i);
8074     }
8075     return buf;
8076 }
8077 function importPrivateKey(pem) {
8078     const binaryDerString = window.atob(pemContents);
8079     const binaryDer = str2ab(binaryDerString);
8080     return window.crypto.subtle.importKey(
8081         "pkcs8",
8082         binaryDer,
8083         {
8084             name: "RSA-PSS",
8085             modulusLength: 2048,
8086             publicExponent: new Uint8Array([1, 0, 1]),
8087             hash: "SHA-256",
8088         },
8089         true,
8090         ["sign"]
8091     );
8092 }
8093 //importPrivateKey(ppem)
8094
8095 //key = {}
8096 //buf = "abc"
8097 //enc = "xyzxxxxxx"; //crypto.publicEncrypt(key,buf)
8098 //b64 = btoa(enc)
8099 //dec = atob(b64)
8100 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8101 </script>
8102 */
8103
8104 /*
8105 <!-- ----- GJConsole BEGIN { ----- -->
8106 <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
8107 <details><summary>GJ Console</summary>
8108 <p>
8109 <span id="GJE_RootNode0"></span>
8110 </p>
8111 <style id="GJConsoleStyle">
8112 .GJConsole {
8113     z-index:1000;
8114     width:400; height:200px;
8115     margin:2px;
8116     color:#fff; background-color:#66a;
8117     font-size:12px; font-family:monospace,Courier New;
8118 }
8119 </style>
8120
8121 <script id="GJConsoleScript" class="GJConsole">
8122     var PS1 = "% "
8123     function GJC_KeyDown(keyevent){
8124         key = keyevent.code
8125         if( key == "Enter" ){
8126             GJC_Command(this)
8127             this.value += "\n" + PS1 // prompt
8128         }else
8129         if( key == "Escape"){
8130             SuppressGJShell = false
8131             GshMenu.focus() // should be previous focus
8132         }
8133     }
8134     var GJC_SessionId
8135     function GJC_SetSessionId(){
8136         var xd = new Date()
8137         GJC_SessionId = xd.getTime() / 1000
8138     }
8139     GJC_SetSessionId()
8140     function GJC_Memory(mem,args,text){
8141         argv = args.split(' ')
8142         cmd = argv[0]
8143         argv.shift()
8144         args = argv.join(' ')
8145         ret = ""
8146
8147         if( cmd == 'clear' ){
8148             Permanent.setItem(mem,'')
8149         }else
8150         if( cmd == 'read' ){
8151             ret = Permanent.getItem(mem)
8152         }else
8153         if( cmd == 'save' ){
8154             val = Permanent.getItem(mem)
8155             if( val == null ){ val = "" }
8156             d = new Date()
8157             val += d.getTime()/1000+ "+GJC_SessionId+" "+document.URL+" "+args+"\n"
8158             val += text.value
8159             Permanent.setItem(mem,val)
8160         }else
8161         if( cmd == 'write' ){
8162             val = Permanent.getItem(mem)
8163             if( val == null ){ val = "" }
8164             d = new Date()
8165             val += d.getTime()/1000+ "+GJC_SessionId+" "+document.URL+" "+args+"\n"
8166             Permanent.setItem(mem,val)
8167         }else{
8168             ret = "Commands: write | read | save | clear"
8169         }
8170         return ret
8171     }
8172     // -- 2020-09-14 added TableEditor
8173     var GJE_CurElement = null; //GJE_RootNode
8174     GJE_NodeSaved = null
8175     GJE_TableNo = 1
8176     function GJE_StyleKeyCommand(kev){
8177         keycode = kev.code
8178         console.log('GJE-Key: '+keycode)
8179         if( keycode == 'Escape' ){
8180             GJE_SetStyle(this);
8181         }
8182         kev.stopPropagation()
8183         // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
8184     }

```

```

8185 var GJE_CommandMode = false
8186 function GJE_TableKeyCommand(kev,tab){
8187   wasCmdMode = GJE_CommandMode
8188   key = kev.code
8189   if( key == 'Escape' ){
8190     console.log("To command mode: "+tab.nodeName+"#"+tab.id)
8191     //tab.setAttribute('contenteditable', false)
8192     tab.style.caretColor = "blue"
8193     GJE_CommandMode = true
8194   }else
8195   if( key == "KeyA" ){
8196     tab.style.caretColor = "red"
8197     GJE_CommandMode = false
8198   }else
8199   if( key == "KeyI" ){
8200     tab.style.caretColor = "red"
8201     GJE_CommandMode = false
8202   }else
8203   if( key == "KeyO" ){
8204     tab.style.caretColor = "red"
8205     GJE_CommandMode = false
8206   }else
8207   if( key == "KeyJ" ){
8208     console.log("ROW-DOWN")
8209   }else
8210   if( key == "KeyK" ){
8211     console.log("ROW-UP")
8212   }else
8213   if( key == "Keyw" ){
8214     console.log("COL-FORW")
8215   }else
8216   if( key == "Keyb" ){
8217     console.log("COL-BACK")
8218   }
8219   kev.stopPropagation()
8220   if( wasCmdMode ){
8221     kev.preventDefault()
8222   }
8223 }
8224
8225 function GJE_DragEvent(ev,elem){
8226   x = ev.clientX
8227   y = ev.clientY
8228   console.log("Dragged: "+this.nodeName+"#"+this.id+" x="+x+" y="+y)
8229 }
8230 // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
8231 // https://www.w3.org/TR/uevents/#events-mouseevents
8232 function GJE_DropEvent(ev,elem){
8233   x = ev.clientX
8234   y = ev.clientY
8235   this.style.x = x
8236   this.style.y = y
8237   this.style.position = 'absolute' // 'fixed'
8238   this.parentNode = gsh // just for test
8239   console.log("Dropped: "+this.nodeName+"#"+this.id+" x="+x+" y="+y
8240     + " parent="+this.parentNode.id)
8241 }
8242 function GJE_SetTableStyle(ev){
8243   this.innerHTML = this.value; // sync. for external representation?
8244   if(false){
8245     stid = this.parentNode.id+this.id
8246     // and remove "span" at the end
8247     e = document.getElementById(stid)
8248     //alert('SetTableStyle #' +e.id+'\n'+this.value)
8249     if( e != null ){
8250       e.innerHTML = this.value
8251     }else{
8252       console.log('Style Not found: '+stid)
8253     }
8254     //alert('event StopPropagation: '+ev)
8255   }
8256 }
8257 function setCSSofClass(cclass,cstyle){
8258   const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
8259   rlen = ss.cssRules.length;
8260   let tabrule = null;
8261   rulex = -1
8262
8263   // should skip white space at the top of cstyle
8264   sel = cstyle.charAt(0);
8265   selector = sel+cclass;
8266   console.log('-- search style rule for '+selector)
8267
8268   for(let i = 0; i < rlen; i++){
8269     cr = ss.cssRules[i];
8270     console.log("CSS rule ['+i+'/' +rlen+''] '+cr.selectorText);
8271     if( cr.selectorText == selector ){ // css class selector
8272       tabrule = ss.cssRules[i];
8273       console.log("CSS rule found for:['+i+'/' +rlen+''] '+selector);
8274       ss.deleteRule(i);
8275       //rlen = ss.cssRules.length;
8276       rulex = i
8277       // should search and replace the property here
8278     }
8279   }
8280   // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
8281   if( tabrule == null ){
8282     console.log("CSS rule NOT found for:['+rlen+''] '+selector);
8283     ss.insertRule(cstyle,rlen);
8284     ss.insertRule(cstyle,0); // override by 0?
8285     console.log("CSS rule inserted:['+(rlen+1)+'']\n"+cstyle);
8286   }else{
8287     ss.insertRule(cstyle,rlen);
8288     ss.insertRule(cstyle,0);
8289     console.log("CSS rule replaced:['+(rlen+1)+'']\n"+cstyle);
8290   }
8291 }
8292 function GJE_SetStyle(te){
8293   console.log('Apply the style to:'+te.id+'\n');
8294   console.log('Apply the style to:'+te.parentNode.id+'\n');
8295   console.log('Apply the style to:'+te.parentNode.class+'\n');
8296   cclass = te.parentNode.class;
8297   setCSSofClass(cclass,te.value); // should get selector part from
8298   // selector { rules }
8299
8300   if(false){
8301     //console.log('Apply the style:')
8302     //stid = this.parentNode.id+this.id+"
8303     //stid = this.id+".style"
8304     css = te.value
8305     stid = te.parentNode.id+".style"
8306     e = document.getElementById(stid)
8307     if( e != null ){
8308       //console.log('Apply the style:'+e.id+'\n'+te.value);

```

```

8309     console.log('Apply the style:'+e.id+'\n'+css);
8310 //     e.innerHTML = css; //te.value;
8311 //ncss = e.sheet;
8312 //ncss.insertRule(te.value,ncss.cssRules.length);
8313 }else{
8314     console.log('No element to Apply the style: '+stid)
8315 }
8316 tblid = te.parentNode.id+".table";
8317 e = document.getElementById(tblid);
8318 if( e != null ){
8319     //e.setAttribute('style',css);
8320     e.setProperty('style',css,'!important');
8321 }
8322 }
8323 }
8324 function makeTable(argv){
8325     //tid = ''
8326     cwe = GJE_CurElement
8327     tid = 'table_' + GJE_TableNo
8328
8329     nt = new Text('\n')
8330     cwe.appendChild(nt)
8331
8332     ne = document.createElement('span'); // the container
8333     cwe.appendChild(ne)
8334     ne.id = tid + '-span'
8335     ne.setAttribute('contenteditable',true)
8336
8337     htspan = document.createElement('span'); // html part
8338     //htspan.id = tid + '-html'
8339     //ne.innerHTML = '\n'
8340     nt = new Text('\n')
8341     ne.appendChild(nt)
8342     ne.appendChild(htspan)
8343
8344     htspan.id = tid
8345     htspan.setAttribute('class',tid)
8346
8347     ne.setAttribute('draggable','true')
8348     ne.addEventListener('drag',GJE_DragEvent);
8349     ne.addEventListener('dragend',GJE_DropEvent);
8350
8351     var col = 3
8352     var row = 2
8353     if( argv[0] != null ){
8354         col = argv[0]
8355         argv.shift()
8356     }
8357     if( argv[0] != null ){
8358         row = argv[0]
8359         argv.shift()
8360     }
8361     //ne.setAttribute('class',tid)
8362     ht = '\n'
8363     //ht += '<'+table ' + 'id="'+tid+'"' + ' class="'+tid+'"'
8364     ht += '<'+table '
8365     + ' onkeydown="GJE_TableKeyCommand(event,this)"'
8366     //+ ' ondrag="GJE_DragEvent(event,this)"\n'
8367     //+ ' ondragend="GJE_DropEvent(event,this)"\n'
8368     //+ ' draggable="true"\n'
8369     //+ ' contenteditable="true"'
8370     + '>\n'
8371     ht += '<'+tbody>\n';
8372     for( r = 0; r < row; r++ ){
8373         ht += "<"+tr>\n"
8374         for( c = 0; c < col; c++ ){
8375             ht += "<"+td>"
8376             ht += " ABCDEFGHIJKLMNOPQRSTUVWXYZ".charAt(c) + r
8377             ht += "<"+tr>\n"
8378         }
8379         ht += "<"+tr>\n"
8380     }
8381     ht += '<'+tbody>\n';
8382     ht += '<'+table>\n';
8383     htspan.innerHTML = ht;
8384     nt = new Text('\n')
8385     ne.appendChild(nt)
8386
8387     st = '#'+tid+' *{\n' // # for instanse specific
8388     + 'border:1px solid #aaa;\n'
8389     + 'background-color:#efe;\n'
8390     + 'color:#222;\n'
8391     + 'font-size:#14pt !important;\n'
8392     + 'font-family:monospace,Courier New !important;\n'
8393     + '}' /* hit ESC to apply */'\n'
8394
8395     // wish script to be included
8396     //nj = document.createElement('script')
8397     //ne.appendChild(nj)
8398     //ne.innerHTML = 'function SetStyle(e){'
8399
8400     // selector seems lost in dynamic style appending
8401     if(false){
8402         ns = document.createElement('style')
8403         ne.appendChild(ns)
8404         ns.id = tid + '.style'
8405         ns.innerHTML = '\n'+st
8406         nt = new Text('\n')
8407         ne.appendChild(nt)
8408     }
8409
8410     setCSSofClass(tid,st); // should be in JavaScript script?
8411
8412     nx = document.createElement('textarea')
8413     ne.appendChild(nx)
8414     nx.id = tid + '-style_def'
8415     nx.setAttribute('class','GJ_StyleEditor')
8416     nx.spellcheck = false
8417     nx.cols = 60
8418     nx.rows = 10
8419     nx.innerHTML = '\n'+st
8420     nx.addEventListener('change',GJE_SetTableStyle);
8421     nx.addEventListener('keydown',GJE_StyleKeyCommand);
8422     //nx.addEventListener('click',GJE_SetTableStyle);
8423
8424     nt = new Text('\n')
8425     cwe.appendChild(nt)
8426
8427     GJE_TableNo += 1
8428     return 'created TABLE id="'+tid+'"'
8429 }
8430 function GJE_NodeEdit(argv){
8431     cwe = GJE_CurElement
8432     cmd = argv[0]

```

```

8433     argv.shift()
8434     args = argv.join(' ')
8435     ret = ""
8436
8437     if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
8438         if( GJE_NodeSaved != null ){
8439             xn = GJE_RootNode
8440             GJE_RootNode = GJE_NodeSaved
8441             GJE_NodeSaved = xn
8442             ret = '-- did undo'
8443         }else{
8444             ret = '-- could not undo'
8445         }
8446     }
8447     return ret
8448 }
8449 GJE_NodeSaved = GJE_RootNode.cloneNode()
8450 if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
8451     if( argv[0] == null ){
8452         ne = GJE_RootNode
8453     }else
8454     if( argv[0] == '..' ){
8455         ne = cwe.parentNode
8456     }else{
8457         ne = document.getElementById(argv[0])
8458     }
8459     if( ne != null ){
8460         GJE_CurElement = ne
8461         ret = "-- current node: " + ne.id
8462     }else{
8463         ret = "-- not found: " + argv[0]
8464     }
8465 }else
8466 if( cmd == '.mkt' || cmd == '.mktable' ){
8467     makeTable(argv)
8468 }else
8469 if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
8470     ne = document.createElement(argv[0])
8471     //ne.id = argv[0]
8472     ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
8473     cwe.appendChild(ne)
8474     if( cmd == '.m' || cmd == '.mk' ){
8475         GJE_CurElement = ne
8476     }
8477 }else
8478 if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
8479     cwe.id = argv[0]
8480 }else
8481 if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
8482 }else
8483 if( cmd == '.h' || cmd == '.sh' || cmd == 'sh' ){
8484     s = argv.join(' ')
8485     cwe.innerHTML = s
8486 }else
8487 if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
8488     cwe.setAttribute(argv[0],argv[1])
8489 }else
8490 if( cmd == '.l' ){
8491 }else
8492 if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
8493     ret = cwe.innerHTML
8494 }else
8495 if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
8496     ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
8497     for( we = cwe.parentNode; we != null; ){
8498         ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
8499         we = we.parentNode
8500     }
8501 }else
8502 {
8503     ret = "Command: mk | rm \n"
8504     ret += "    pw -- print current node\n"
8505     ret += "    mk type -- make node with name and type\n"
8506     ret += "    nm name -- set the id #name of current node\n"
8507     ret += "    rm name -- remove named node\n"
8508     ret += "    cd name -- change current node\n"
8509 }
8510 //alert(ret)
8511 return ret
8512 }
8513 function GJC_Command(text){
8514     lines = text.value.split('\n')
8515     line = lines[lines.length-1]
8516     argv = line.split(' ')
8517     text.value += '\n'
8518     if( argv[0] == '%' ){ argv.shift() }
8519     args0 = argv.join(' ')
8520     cmd = argv[0]
8521     argv.shift()
8522     args = argv.join(' ')
8523
8524     if( cmd == 'nolog' ){
8525         StopConsoleLog = true
8526     }else
8527     if( cmd == 'new' ){
8528         if( argv[0] == 'table' ){
8529             argv.shift()
8530             console.log('argv'+argv)
8531             text.value += makeTable(argv)
8532         }else
8533         if( argv[0] == 'console' ){
8534             text.value += GJ_NewConsole('GJ_Console')
8535         }else{
8536             text.value += '-- new { console | table }'
8537         }
8538     }else
8539     if( cmd == 'strip' ){
8540         //text.value += GJF_StripeClass()
8541     }else
8542     if( cmd == 'css' ){
8543         sel = '#table_1'
8544         if(argv[0]=='0')
8545             rule1 = sel+'{color:#000 !important; background-color:#fff !important;}';
8546         else
8547             rule1 = sel+'{color:#E00 !important; background-color:#eef !important;}';
8548         document.styleSheets[3].deleteRule(0);
8549         document.styleSheets[3].insertRule(rule1,0);
8550         text.value += 'CSS rule added: '+rule1
8551     }else
8552     if( cmd == 'print' ){
8553         e = null;
8554         if( e == null ){
8555             e = document.getElementById('GJFactory_0')
8556         }
8557         if( e == null ){

```

```

8557     e = document.getElementById( 'GJFactory_1' )
8558   }
8559   if( argv[0] != null ){
8560     id = argv[0]
8561     if( id == 'f' ){
8562       //e = document.getElementById( 'GJE_RootNode' );
8563     }else{
8564       e = document.getElementById(id)
8565     }
8566     if( e != null ){
8567       text.value += e.outerHTML
8568     }else{
8569       text.value += "Not found: " + id
8570     }
8571   }else{
8572     text.value += GJE_RootNode.outerHTML
8573     //text.value += e.innerHTML
8574   }
8575   }else
8576   if( cmd == 'destroy' ){
8577     text.value += GJFactory_Destroy()
8578   }else
8579   if( cmd == 'save' ){
8580     e = document.getElementById( 'GJFactory' )
8581     Permanent.setItem( 'GJFactory-1', e.innerHTML )
8582     text.value += "-- Saved GJFactory"
8583   }else
8584   if( cmd == 'load' ){
8585     gjf = Permanent.getItem( 'GJFactory-1' )
8586     e = document.getElementById( 'GJFactory' )
8587     e.innerHTML = gjf
8588     // must restore EventListener
8589     text.value += "-- EventListener was not restored"
8590   }else
8591   if( cmd.charAt(0) == '.' ){
8592     argv0 = argv0.split( '.' )
8593     text.value += GJE_NodeEdit( argv0 )
8594   }else
8595   if( cmd == 'cont' ){
8596     bannerIsStopping = false
8597     GshMenuStop.innerHTML = "Stop"
8598   }else
8599   if( cmd == 'date' ){
8600     text.value += DateLong()
8601   }else
8602   if( cmd == 'echo' ){
8603     text.value += args
8604   }else
8605   if( cmd == 'fork' ){
8606     html_fork()
8607   }else
8608   if( cmd == 'last' ){
8609     text.value += MyHistory
8610     //h = document.createElement( "span" )
8611     //h.innerHTML = MyHistory
8612     //text.value += h.innerHTML
8613     //tx = MyHistory.replace( "\n", "" )
8614     //text.value += tx.replace( "<"+>",">"\n" ) + "xxxx<"+>yyyy"
8615   }else
8616   if( cmd == 'ne' ){
8617     text.value += GJE_NodeEdit( argv )
8618   }else
8619   if( cmd == 'reload' ){
8620     location.reload()
8621   }else
8622   if( cmd == 'mem' ){
8623     text.value += GJC_Memory( 'GJC_Storage', args, text )
8624   }else
8625   if( cmd == 'stop' ){
8626     bannerIsStopping = true
8627     GshMenuStop.innerHTML = "Start"
8628   }else
8629   if( cmd == 'who' ){
8630     text.value += "SessionId="+GJC_SessionId+" "+document.URL
8631   }else
8632   if( cmd == 'wall' ){
8633     text.value += GJC_Memory( 'GJC_Wall', 'write', text )
8634   }else
8635   {
8636     text.value += "Commands: help | echo | date | last \n"
8637     + '      new | save | load | mem \n'
8638     + '      who | wall | fork | nife'
8639   }
8640 }
8641
8642 function GJC_Input(){
8643   if( this.value.endsWith( "\n" ) ){ // remove NL added by textarea
8644     this.value = this.value.slice( 0, this.value.length-1 )
8645   }
8646 }
8647
8648 var GJC_Id = null
8649 function GJC_Resize(){
8650   GJC_Id.style.zIndex = 20000
8651   GJC_Id.style.width = window.innerWidth - 16
8652   GJC_Id.style.height = 300
8653   GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
8654   GJC_Id.style.color = "rgba(255,255,255,1.0)"
8655 }
8656 function GJC_FocusIn(){
8657   this.spellcheck = false
8658   SuppressGJShell = true
8659   this.onkeydown = GJC_Keydown
8660   GJC_Resize()
8661 }
8662 function GJC_FocusOut(){
8663   SuppressGJShell = false
8664   this.removeEventListener( 'keydown', GJC_Keydown );
8665 }
8666 window.addEventListener( 'resize', GJC_Resize );
8667
8668 function GJC_OnStorage(e){
8669   //alert( 'Got Message' )
8670   //GJC.value += "\n((ReceivedMessage))\n"
8671 }
8672 window.addEventListener( 'storage', GJC_OnStorage );
8673 //window.addEventListener( 'storage', ()=>{alert( 'GotMessage' )} )
8674
8675 function GJC_Setup( gjcId ){
8676   gjcId.style.width = gsh.getBoundingClientRect().width
8677   gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
8678   //gjcId.value += "Date: " + DateLong() + "\n"
8679   gjcId.value += PS1
8680   gjcId.onfocus = GJC_FocusIn

```

```

8681     gjcId.addEventListener('input',GJC_Input);
8682     gjcId.addEventListener('focusout',GJC_FocusOut);
8683     GJC_Id = gjcId
8684 }
8685 function GJC_Clear(id){
8686 }
8687 if( document.getElementById("GJC_0") != null ){
8688     GJC_Setup(GJC_0)
8689 }else{
8690     document.write('<'+`textarea id="GJC_1" class="GJConsole"><'+`/textarea>')
8691     GJC_Setup(GJC_1)
8692     factory = document.createElement('span');
8693     gsh.appendChild(factory)
8694     GJE_RootNode = factory;
8695     GJE_CurElement = GJE_RootNode;
8696 }
8697
8698 // TODO: focus handling
8699 </script>
8700 <style>
8701 .GJ_StyleEditor {
8702     font-size:9pt !important;
8703     font-family:Courier New, monospace !important;
8704 }
8705 </style>
8706
8707 </details>
8708 </span>
8709 <!-- ----- GJConsole END ) ----- -->
8710 */
8711
8712 /*
8713 <span id="BlinderText">
8714 <style id="BlinderTextStyle">
8715 #GJLinkView {
8716     xxposition:absolute; z-index:5000;
8717     position:relative;
8718     display:block;
8719     left:8px;
8720     color:#fff;
8721     width:800px; height:300px; resize:both;
8722     margin:0px; padding:4px;
8723     background-color:rgba(200,200,200,0.5) !important;
8724 }
8725 .MssgText {
8726     width:578px !important;
8727     resize:both !important;
8728     color:#000 !important;
8729 }
8730 .GjNote {
8731     font-family:Georgia !important;
8732     font-size:13pt !important;
8733     color:#22a !important;
8734 }
8735 .textField {
8736     display:inline;
8737     border:0.5px solid #444;
8738     border-radius:3px;
8739     color:#000; background-color:#fff;
8740     width:106pt; height:18pt;
8741     margin:2px;
8742     padding:2px;
8743     resize:none;
8744     vertical-align:middle;
8745     font-size:10pt; font-family:Courier New;
8746 }
8747 .textLabel {
8748     border:0px solid #000 !important;
8749     background-color:rgba(0,0,0,0);
8750 }
8751 .textURL {
8752     width:300pt !important;
8753     border:0px solid #000 !important;
8754     background-color:rgba(0,0,0,0);
8755 }
8756 .VisibleText {
8757 }
8758 .BlinderText {
8759     color:#000; background-color:#eee;
8760 }
8761 .joinButton {
8762     font-family:Georgia !important;
8763     font-size:11pt;
8764     line-height:1.1;
8765     height:18pt;
8766     width:50pt;
8767     padding:3px !important;
8768     text-align:center !important;
8769     border-color:#aaa !important;
8770     border-radius:5px;
8771     color:#fff; background-color:#4a4 !important;
8772     vertical-align:middle !important;
8773 }
8774 .SendButton {
8775     vertical-align:top;
8776 }
8777 .ws0_log {
8778     font-size:10pt;
8779     color:#000 !important;
8780     line-height:1.0;
8781     background-color:rgba(255,255,255,0.7) !important;
8782     font-family:Courier New,monospace !important;
8783     width:99.3%;
8784     white-space:pre;
8785 }
8786 </style>
8787
8788 <!-- Form autofill test
8789 Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafm/formLogin" size="80">
8790 <form method="POST" id="xxform" action="https://192.168.10.1/boafm/formLogin">
8791 dest? <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
8792 -->
8793 <details><summary>Form Auto. Filling</summary>
8794 <style>
8795 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
8796     display:inline !important; font-size:10pt !important; padding:1px !important;
8797 }
8798 </style>
8799 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
8800 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
8801 Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
8802 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
8803 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
8804 Sessionid:<input id="xxssid" class="xxinput" name="SESSION_ID" type="text" size="80">

```

```

8805     <input id="xxsubmit" class="xxinput" type="submit" value="Submit"></form>
8806 </span>
8807 <script>
8808     function XXSetFormAction(){
8809         xxform.setAttribute('action',xxserv.value);
8810     }
8811     xxform.setAttribute('action',xxserv.value);
8812     xxserv.addEventListener('change',XXSetFormAction);
8813     //xxserv.value = location.href;
8814 </script>
8815 </details>
8816 */
8817
8818 /*
8819 <details id="BlinderTextClass" class="gsh-src"><summary>BlinderText</summary>
8820 <span id="BlinderTextScript">
8821     // https://w3c.github.io/uievents/#event-type-keydown
8822     //
8823     // 2020-09-21 class BlinderText - textarea element not to be readable
8824     //
8825     // BlinderText attributes
8826     // bl_plainText - null
8827     // bl_hideChecksum - [false]
8828     // bl_showLength - [false]
8829     // bl_visible - [false]
8830     // data-bl_config - []
8831     // - min. length
8832     // - max. length
8833     // - acceptable charset in generate text
8834     //
8835     function BlinderChecksum(text){
8836         plain = text.bl_plainText;
8837         return strCRC32(plain,plain.length).toFixed(0);
8838     }
8839     function BlinderKeydown(ev){
8840         pass = ev.target
8841         if( ev.code == 'Enter' ){
8842             ev.preventDefault();
8843         }
8844         ev.stopPropagation()
8845     }
8846     function BlinderKeyUp1(ev){
8847         blind = ev.target
8848         if( ev.code == 'Backspace'){
8849             blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
8850         }else
8851         if( and(ev.code == 'KeyV', ev.ctrlKey) ){
8852             blind.bl_visible = !blind.bl_visible;
8853         }else
8854         if( and(ev.code == 'KeyL', ev.ctrlKey) ){
8855             blind.bl_showLength = !blind.bl_showLength;
8856         }else
8857         if( and(ev.code == 'KeyU', ev.ctrlKey) ){
8858             blind.bl_plainText = "";
8859         }else
8860         if( and(ev.code == 'KeyR', ev.ctrlKey) ){
8861             checksum = BlinderChecksum(blind);
8862             blind.bl_plainText = checksum; //$.toString(32);
8863         }else
8864         if( ev.code == 'Enter' ){
8865             ev.stopPropagation();
8866             ev.preventDefault();
8867             return;
8868         }else
8869         if( ev.key.length != 1 ){
8870             console.log('KeyUp: '+ev.code+'/'+ev.key);
8871             return;
8872         }else{
8873             blind.bl_plainText += ev.key;
8874         }
8875
8876         leng = blind.bl_plainText.length;
8877         //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
8878         checksum = BlinderChecksum(blind) % 10; // show last one digit only
8879
8880         visual = '';
8881         if( !blind.bl_hideChecksum || blind.bl_showLength ){
8882             visual += '[';
8883         }
8884         if( !blind.bl_hideChecksum ){
8885             visual += '#' + checksum.toString(10);
8886         }
8887         if( blind.bl_showLength ){
8888             visual += '/' + leng;
8889         }
8890         if( !blind.bl_hideChecksum || blind.bl_showLength ){
8891             visual += ' ';
8892         }
8893         if( blind.bl_visible ){
8894             visual += blind.bl_plainText;
8895         }else{
8896             visual += '*'.repeat(leng);
8897         }
8898         blind.value = visual;
8899     }
8900     function BlinderKeyUp(ev){
8901         BlinderKeyUp1(ev);
8902         ev.stopPropagation();
8903     }
8904     // https://w3c.github.io/uievents/#keyboardevent
8905     // https://w3c.github.io/uievents/#uievent
8906     // https://dom.spec.whatwg.org/#event
8907     function BlinderTextEvent(){
8908         ev = event;
8909         blind = ev.target;
8910         console.log('Event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
8911         if( ev.type == 'keyup' ){
8912             BlinderKeyUp(ev);
8913         }else
8914         if( ev.type == 'keydown' ){
8915             BlinderKeydown(ev);
8916         }else{
8917             console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
8918         }
8919     }
8920     //< textarea hidden id="BlinderTextClassDef" class="textField"
8921     // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
8922     // spellcheck="false">< /textarea>
8923     //< textarea hidden id="gj_pass1"
8924     // class="textField BlinderText"
8925     // placeholder="PassWord1"
8926     // onkeydown="BlinderTextEvent()"
8927     // onkeyup="BlinderTextEvent()"
8928     // spellcheck="false"> /textarea>

```



```

8929 function SetupBlinderText(parent,txa,phold){
8930     if( txa == null ){
8931         txa = document.createElement('textarea');
8932         //txa.id = id;
8933     }
8934     txa.setAttribute('class','textField BlinderText');
8935     txa.setAttribute('placeholder',phold);
8936     txa.setAttribute('onkeydown','BlinderTextEvent()');
8937     txa.setAttribute('onkeyup','BlinderTextEvent()');
8938     txa.setAttribute('spellcheck','false');
8939     //txa.setAttribute('bl_plainText','false');
8940     txa.bl_plainText = '';
8941     //parent.appendChild(txa);
8942 }
8943 function DestroyBlinderText(txa){
8944     txa.removeAttribute('class');
8945     txa.removeAttribute('placeholder');
8946     txa.removeAttribute('onkeydown');
8947     txa.removeAttribute('onkeyup');
8948     txa.removeAttribute('spellcheck');
8949     txa.bl_plainText = '';
8950 }
8951 //
8952 // visible textarea like Username
8953 //
8954 function VisibleTextEvent(){
8955     if( event.code == 'Enter' ){
8956         if( event.target.NoEnter ){
8957             event.preventDefault();
8958         }
8959     }
8960     event.stopPropagation();
8961 }
8962 function SetupVisibleText(parent,txa,phold){
8963     if( false ){
8964         txa.setAttribute('class','textField VisibleText');
8965     }else{
8966         newclass = txa.getAttribute('class');
8967         if( and(newclass != null, newclass != '') ){
8968             newclass += ' ';
8969         }
8970         newclass += 'VisibleText';
8971         txa.setAttribute('class',newclass);
8972     }
8973     //console.log('SetupVisibleText class='+txa.class);
8974     txa.setAttribute('placeholder',phold);
8975     txa.setAttribute('onkeydown','VisibleTextEvent()');
8976     txa.setAttribute('onkeyup','VisibleTextEvent()');
8977     txa.setAttribute('spellcheck','false');
8978     cols = txa.getAttribute('cols');
8979     if( cols != null ){
8980         txa.style.width = '580px';
8981         //console.log('VisualText#'+txa.id+' cols='+cols)
8982     }else{
8983         //console.log('VisualText#'+txa.id+' NO cols')
8984     }
8985     rows = txa.getAttribute('rows');
8986     if( rows != null ){
8987         txa.style.height = '30px';
8988         txa.style.resize = 'both';
8989         txa.NoEnter = false;
8990     }else{
8991         txa.NoEnter = true;
8992     }
8993 }
8994 function DestroyVisibleText(txa){
8995     txa.removeAttribute('class');
8996     txa.removeAttribute('placeholder');
8997     txa.removeAttribute('onkeydown');
8998     txa.removeAttribute('onkeyup');
8999     txa.removeAttribute('spellcheck');
9000     cols = txa.removeAttribute('cols');
9001 }
9002 </span>
9003 <script>
9004 js = document.getElementById('BlinderTextScript');
9005 eval(js.innerHTML);
9006 //js.outerHTML = ""
9007 </script>
9008
9009 </details>
9010 </span>
9011 */
9012
9013 /*
9014 <script id="GJLinkScript">
9015 function gjkey_hash(text){
9016     return strCRC32(text,text.length) % 0x10000;
9017 }
9018 function gj_addlog(e,msg){
9019     now = (new Date().getTime() / 1000).toFixed(3);
9020     tstp = '['+now+' ] '
9021     e.value += tstp + msg;
9022     e.scrollTop = e.scrollHeight;
9023 }
9024 function gj_addlog_cl(msg){
9025     ws0_log.value += '(console.log) ' + msg + '\n';
9026 }
9027 var GJ_Channel = null;
9028 var GJ_Log = null;
9029 var gjx; // the global variable
9030 function GJ_Join(){
9031     target = gj_join;
9032     if( target.value == 'Leave' ){
9033         GJ_Channel.close();
9034         GJ_Channel = null;
9035         target.value = 'Join';
9036         return;
9037     }
9038
9039     var ws0;
9040     var ws0_log;
9041
9042     sav_console_log = console.error
9043     console.error = gj_addlog_cl
9044     ws0 = new WebSocket(gj_serv.innerHTML);
9045     console.error = sav_console_log
9046
9047     GJ_Channel = ws0;
9048     ws0_log = document.getElementById('ws0_log');
9049     GJ_Log = ws0_log;
9050
9051     now = (new Date().getTime() / 1000).toFixed(3);
9052     const wsstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];

```

```

9053 cst = wsstats[ws0.readyState];
9054 gj_addlog(ws0_log,'stat '+ws0.readyState+'('+cst+') : GJ Linked\n');
9055
9056 ws0.addEventListener('error', function(event){
9057   gj_addlog(ws0_log,'stat error : transport error?\n');
9058 });
9059 ws0.addEventListener('open', function(event){
9060   GJLinkView.style.zIndex = 10000;
9061   //console.log('#'+GJLinkView.id+'.zIndex='+GJLinkView.style.zIndex);
9062   date1 = new Date().getTime();
9063   date2 = (date1 / 1000).toFixed(3);
9064   seed = date1.toString(16);
9065
9066   // user name and key
9067   user = document.getElementById('gj_user').value;
9068   if( user.length == 0 ){
9069     gj_user.value = 'nemo';
9070     user = 'nemo';
9071   }
9072   key1 = document.getElementById('gj_ukey').bl_plainText;
9073   ukey = gjkey_hash(seed+user+key1).toString(16);
9074
9075   // session name and key
9076   chan = document.getElementById('gj_chan').value;
9077   if( chan.length == 0 ){
9078     gj_chan.value = 'main';
9079     chan = 'main';
9080   }
9081   key2 = document.getElementById('gj_ckey').bl_plainText;
9082   ckey = gjkey_hash(seed+chan+key2).toString(16);
9083
9084   msg = date2 + ' JOIN ' + user + '|' + chan + ' ' + ukey + ':' + ckey;
9085   gj_addlog(ws0_log,'send '+msg+'\n');
9086   ws0.send(msg);
9087
9088   target.value = 'Leave';
9089   //console.log([''+date2+''] #'+target.id+' '+target.value+'\n');
9090   //gj_addlog(ws0_log,'label '+target.value+'\n');
9091 });
9092 ws0.addEventListener('message', function(event){
9093   now = (new Date().getTime() / 1000).toFixed(3);
9094   msg = event.data;
9095   gj_addlog(ws0_log,'recv '+msg+'\n');
9096
9097   argv = msg.split(' ');
9098   tstamp = argv[0];
9099   argv.shift();
9100   if( argv[0] == 'reload' ){
9101     location.reload()
9102   }
9103   argv.shift(); // command
9104   argv.shift(); // from/to
9105   if( argv[0] == 'auth' ){
9106     // doing authorization required
9107   }
9108   if( argv[0] == 'echo' ){
9109     now = (new Date().getTime() / 1000).toFixed(3);
9110     msg = now+' '+RESP +argv.join(' ');
9111     gj_addlog(ws0_log,'send '+msg+'\n');
9112     ws0.send(msg);
9113   }
9114   if( argv[0] == 'eval' ){
9115     argv.shift();
9116     js = argv.join(' ');
9117     ret = eval(js); // <----- eval()
9118     gj_addlog(ws0_log,'eval '+js+' = '+ret+'\n');
9119     now = (new Date().getTime() / 1000).toFixed(3);
9120     msg = now + ' ' + RESP + ret;
9121     ws0.send(msg);
9122     gj_addlog(ws0_log,'send '+msg+'\n')
9123   }
9124 });
9125 ws0.addEventListener('close', function(event){
9126   if( GJ_Channel == null ){
9127     gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
9128     return;
9129   }
9130   GJ_Channel.close();
9131   GJ_Channel = null;
9132   target.value = 'Join';
9133   gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
9134 });
9135 }
9136 function GJ_SendMessageUserPass(user,chan,msgbody){
9137   now = (new Date().getTime() / 1000).toFixed(3);
9138   msg = now + ' ISAY '+user+'|'+chan+' '+ msgbody;
9139   gj_addlog(GJ_Log,'send '+msg+'\n');
9140   GJ_Channel.send(msg);
9141 }
9142 function GJ_SendMessage(msgbody){
9143   if( GJ_Channel == null ){
9144     gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
9145     return;
9146   }
9147   //target = event.target;
9148   user = document.getElementById('gj_user').value;
9149   chan = document.getElementById('gj_chan').value;
9150   GJ_SendMessageUserPass(user,chan,msgbody);
9151 }
9152 function GJ_Send(){
9153   msgbody = gj_sendText.value;
9154   GJ_SendMessage(msgbody);
9155 }
9156 </script>
9157
9158 <!-- ----- GJLINK ----- -->
9159 <!--
9160 - User can subscribe to a channel
9161 - A channel will be broadcasted
9162 - A channel can be a pattern (regular expression)
9163 - User is like From:(me) and channel is like To: or Recipient:
9164 - like VIABUS
9165   - watch message with SENDME, WATCH, CATCH, HEAR, or so
9166   - routing with path expression or name pattern (with routing with DNS like system)
9167 -->
9168 */
9169
9170 </span id="GJLinkGolang">
9171 <details id="GshWebSocket" class="gsh-src"><summary>Golang / JavaScript Link</summary>
9172 // 2020-0920 created
9173 <a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
9174 <a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
9175 // INSTALL: go get golang.org/x/net/websocket
9176 // INSTALL: sudo {apt,yum} install git (if git is not instilled yet)

```

```

9177 // import "golang.org/x/net/websocket"
9178 const gshws_origin = "http://localhost:9999"
9179 const gshws_server = "localhost:9999"
9180 const gshws_port = 9999
9181 const gshws_path = "gmlink"
9182 const gshws_url = "ws://" + gshws_server + "/" + gshws_path
9183 const GSHWS_MSGSIZE = (8*1024)
9184 func fmtstring(fmts string, params ...interface{})(string){
9185     return fmt.Sprintf(fmts,params...)
9186 }
9187 func GSHWS_MARK(what string)(string){
9188     now := time.Now()
9189     us := fmtstring("%06d",now.Nanosecond() / 1000)
9190     mark := ""
9191     if( !AtConsoleLineTop ){
9192         mark += "\n"
9193         AtConsoleLineTop = true
9194     }
9195     mark += "[" + now.Format(time.Stamp) + "." + us + "]" -GJ- + what + ": "
9196     return mark
9197 }
9198 func gchk(what string,err error){
9199     if( err != nil ){
9200         panic(GSHWS_MARK(what)+err.Error())
9201     }
9202 }
9203 func glog(what string, fmts string, params ...interface{}){
9204     fmt.Print(GSHWS_MARK(what))
9205     fmt.Printf(fmts+"\n",params...)
9206 }
9207
9208 var WSV = []*websocket.Conn{}
9209 func jsend(argv []string){
9210     if len(argv) <= 1 {
9211         fmt.Printf("--Ij %v [-m] command arguments\n",argv[0])
9212         return
9213     }
9214     argv = argv[1:]
9215     if( len(WSV) == 0 ){
9216         fmt.Printf("--Ej-- No link now\n")
9217         return
9218     }
9219     if( 1 < len(WSV) ){
9220         fmt.Printf("--Ij-- multiple links (%v)\n",len(WSV))
9221     }
9222
9223     multicast := false // should be filtered with regexp
9224     if( 0 < len(argv) && argv[0] == "-m" ){
9225         multicast = true
9226         argv = argv[1:]
9227     }
9228     args := strings.Join(argv, " ")
9229
9230     now := time.Now()
9231     msec := now.UnixNano() / 1000000;
9232     tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9233     msg := fmtstring("%v SEND gshell|+ %v",tstamp,args)
9234
9235     if( multicast ){
9236         for i,ws := range WSV {
9237             wn,werr := ws.Write([]byte(msg))
9238             if( werr != nil ){
9239                 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
9240             }
9241             glog("SQ",fmtstring("(%v) %v",wn,msg))
9242         }
9243     }else{
9244         i := 0
9245         ws := WSV[i]
9246         wn,werr := ws.Write([]byte(msg))
9247         if( werr != nil ){
9248             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
9249         }
9250         glog("SQ",fmtstring("(%v) %v",wn,msg))
9251     }
9252 }
9253 func servl(ws *websocket.Conn) {
9254     WSV = append(WSV,ws)
9255     //fmt.Print("\n")
9256     glog("CO","accepted connections[%v]",len(WSV))
9257     //remoteAddr := ws.RemoteAddr
9258     //fmt.Printf("-- accepted %v\n",remoteAddr)
9259     //fmt.Printf("-- accepted %v\n",ws.Config())
9260     //fmt.Printf("-- accepted %v\n",ws.Config().Header)
9261     //fmt.Printf("-- accepted %v // %v\n",ws,servl)
9262
9263     var reqb = make([]byte,GSHWS_MSGSIZE)
9264     for {
9265         rn, rerr := ws.Read(reqb)
9266         if( rerr != nil || rn < 0 ){
9267             glog("SQ",fmtstring("(%v,%v)",rn,rerr))
9268             break
9269         }
9270         req := string(reqb[0:rn])
9271         glog("SQ",fmtstring("(%v) %v",rn,req))
9272
9273         margv := strings.Split(req, " ");
9274         margv = margv[1:]
9275         if( 0 < len(margv) ){
9276             if( margv[0] == "RESP" ){
9277                 // should forward to the destination
9278                 continue;
9279             }
9280         }
9281         now := time.Now()
9282         msec := now.UnixNano() / 1000000;
9283         tstamp := fmtstring("%.3f",float64(msec)/1000.0)
9284         res := fmtstring("%v "+CAST+" %v",tstamp,req)
9285         wn, werr := ws.Write([]byte(res))
9286         gchk("SE",werr)
9287         glog("SR",fmtstring("(%v) %v",wn,string(res)))
9288     }
9289     glog("SF","WS response finish")
9290
9291     wsv := []*websocket.Conn{}
9292     wxs := 0
9293     for i,v := range WSV {
9294         if( v != ws ){
9295             wxs = i
9296             wsv = append(wsv,v)
9297         }
9298     }
9299     WSV = wsv
9300     //glog("CO","closed %v",ws)

```

```

9301     glog("CO","closed connection [%v/%v]",wsx+1,len(WSV)+1)
9302     ws.Close()
9303 }
9304 // url := [scheme://]host[:port][/path]
9305 func decomp_URL(url string){
9306 }
9307 func full_wsURL(){
9308 }
9309 func gj_server(argv []string) {
9310     gjserv := gshws_url
9311     gjport := gshws_server
9312     gjpath := gshws_path
9313     gjscheme := "ws"
9314
9315     //cmd := argv[0]
9316     argv = argv[1:]
9317     if( 1 <= len(argv) ){
9318         serv := argv[0]
9319         if( 0 < strings.Index(serv, "://") ){
9320             schemev := strings.Split(serv, "://")
9321             gjscheme = schemev[0]
9322             serv = schemev[1]
9323         }
9324         if( 0 < strings.Index(serv, "/") ){
9325             pathv := strings.Split(serv, "/")
9326             serv = pathv[0]
9327             gjpath = pathv[1]
9328         }
9329         servv := strings.Split(serv, ":")
9330         host := "localhost"
9331         port := 9999
9332         if( servv[0] != "" ){
9333             host = servv[0]
9334         }
9335         if( len(servv) == 2 ){
9336             fmt.Sscanf(servv[1], "%d", &port)
9337         }
9338         //glog("LC", "hostport=%v (%v : %v)", servv, host, port)
9339         gjport = fmt.Sprintf("%v:%v", host, port)
9340         gjserv = gjscheme + "://" + gjport + "/" + gjpath
9341     }
9342     glog("LS", fmt.Sprintf("listening at %v", gjserv))
9343     http.Handle("/"+gjpath, websocket.Handler(serv))
9344     err := error(nil)
9345     if( gjscheme == "wss" ){
9346         // https://golang.org/pkg/net/http/#ListenAndServeTLS
9347         //err = http.ListenAndServeTLS(gjport, nil)
9348     }else{
9349         err = http.ListenAndServe(gjport, nil)
9350     }
9351     gchk("LE", err)
9352 }
9353
9354 func gj_client(argv []string) {
9355     glog("CS", fmt.Sprintf("connecting to %v", gshws_url))
9356     ws, err := websocket.Dial(gshws_url, "", gshws_origin)
9357     gchk("C", err)
9358
9359     var resb = make([]byte, GSHWS_MSGSIZE)
9360     for qi := 0; qi < 3; qi++ {
9361         req := fmt.Sprintf("Hello, GShell! (%v)", qi)
9362         wn, werr := ws.Write([]byte(req))
9363         glog("QM", fmt.Sprintf("%v %v", wn, req))
9364         gchk("QE", werr)
9365         rn, rerr := ws.Read(resb)
9366         gchk("RE", rerr)
9367         glog("RM", fmt.Sprintf("%v %v", rn, string(resb)))
9368     }
9369     glog("CF", "WS request finish")
9370 }
9371 //</details></span>
9372
9373 /*
9374 <details><summary>GJ Link</summary>
9375 <span id="GJLinkView" class="GJLinkView">
9376 <p>
9377 <note class="GjNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
9378 </p>
9379 <p>
9380 <span id="GJLink 1">
9381 <script id="gj_xxx1_gen">
9382 if( document.getElementById('gj_serv') == null ){ // executed twice??
9383 document.write('<'+span id="gj_serv_label" class="textField textLabel">Server: <'+span>');
9384 document.write('<'+span id="gj_serv" class="textField textURL" contenteditable><'+span>');
9385 }
9386 </script>
9387 <br>
9388 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
9389 <script id="gj_xxx2_gen">
9390 if( true ){
9391 document.write('<'+textarea id="gj_user" class="textField"><'+/textarea>');
9392 document.write('<'+textarea id="gj_ukey" class="textField"><'+/textarea>');
9393 document.write('<'+textarea id="gj_chan" class="textField"><'+/textarea>');
9394 document.write('<'+textarea id="gj_ckekey" class="textField"><'+/textarea>');
9395 }
9396 </script>
9397 <br>
9398 <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
9399 <script id="gj_sendText_gen">
9400 if( true ){
9401 document.write('<'+textarea id="gj_sendText" class="textField MssgText" cols=60 rows=2><'+/textarea>');
9402 }
9403 </script>
9404 </span></p>
9405 <p>
9406 <script id="ws0_log_gen">
9407 if( true ){
9408 document.write('<'+textarea id="ws0_log" class="ws0_log"
9409 '+ cols=100 rows=10 spellcheck="false"><'+/textarea>');
9410 }
9411 </script>
9412 </p>
9413 </span>
9414 <script>
9415 function SetupGJLink(){
9416     SetupVisibleText(GJLink_1, gj_serv, 'GJLinkSv');
9417     SetupVisibleText(GJLink_1, gj_user, 'UserName');
9418     SetupBlinderText(GJLink_1, gj_ukey, 'UserKey');
9419     SetupVisibleText(GJLink_1, gj_chan, 'ChannelName');
9420     SetupBlinderText(GJLink_1, gj_ckekey, 'ChannelKey');
9421     SetupVisibleText(GJLink_1, gj_sendText, 'Message');
9422     gj_serv.innerHTML = 'ws://localhost:9999/gjlink1'
9423 }
9424 SetupGJLink();

```

```

9425 function iselem(eid){
9426     return document.getElementById(eid);
9427 }
9428 function DestroyGJLinkl(){
9429     if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
9430     if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
9431     if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
9432     if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
9433     if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
9434     if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
9435     if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
9436     if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
9437 }
9438 DestroyGJLink = DestroyGJLinkl;
9439 </script>
9440 </details>
9441 */
9442
9443 /*
9444 <script id="HtmlCodeview-script">
9445     function showHtmlCode(otxa,code){
9446         if( event.target.value == 'ShowCode' ){
9447             txa = document.createElement('textarea');
9448             txa.id = otxa.id;
9449             txa.setAttribute('class','HtmlCodeviewText');
9450             otxa.parentNode.replaceChild(txa,otxa);
9451             txa.setAttribute('spellcheck','false');
9452             txa.value = code.innerHTML;
9453             txa.style.display = "block";
9454             txa.style.width = "100%";
9455             txa.style.height = "300px";
9456             event.target.value = 'HideCode';
9457         }else{
9458             otxa.style.display = "none";
9459             event.target.value = 'ShowCode';
9460         }
9461     }
9462 </script>
9463 <style id="HtmlCodeview-style">
9464     .HtmlCodeviewText {
9465         font-size:10pt;
9466         font-family:Courier New;
9467         white-space:pre;
9468     }
9469     .HtmlCodeViewButton {
9470         padding:2pt !important;
9471         line-height:1.1 !important;
9472         border:2px inset #bbb !important;
9473         font-size:11pt !important;
9474         font-weight:normal !important;
9475         font-family:Georgia !important;
9476         border-radius:3px !important;
9477         color:#ddd; background-color:#228 !important;
9478     }
9479 </style>
9480 */
9481
9482 /*
9483 <details><summary>Live HTML Snapshot</summary>
9484 <span id="LiveHTML">
9485 <!-- ----- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
9486 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLCode()">
9487 <span id="LiveHTML_Codeview"></span>
9488 <script id="LiveHtmlScript">
9489     function showLiveHTMLCode(){
9490         showHtmlCode(LiveHTML_Codeview,LiveHTML);
9491     }
9492     var _editable = false;
9493     var savSuppressGJShell = false;
9494     function ToggleEditMode(){
9495         _editable = !_editable;
9496         if( _editable ){
9497             savSuppressGJShell = SuppressGJShell;
9498             SuppressGJShell = true;
9499             gsh.setAttribute('contenteditable','true');
9500             GshMenuEdit.innerHTML = 'Lock';
9501             GshMenuEdit.style.color = 'rgba(255,0,0,1)';
9502             GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
9503         }else{
9504             SuppressGJShell = savSuppressGJShell;
9505             gsh.setAttribute('contenteditable','false');
9506             GshMenuEdit.innerHTML = 'Edit';
9507             GshMenuEdit.style.color = 'rgba(16,160,16,1)';
9508             GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
9509         }
9510     }
9511     function html_edit(){
9512         ToggleEditMode();
9513     }
9514
9515 // Live HTML (DOM) Snapshot onto browser's localStorage
9516 // 2020-0923 SatoxITS
9517 var htRoot = gsh // -- Element-ID, should be selectable
9518 const snappedHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStogate
9519 // -- should be a [map] of URL
9520 // -- should be with CSSOM as inline script
9521 const htVersionTag = 'VersionTag'; // VesionTag Element-ID in the HTML (in DOM)
9522 function showVersion(note,w,v,u,t){
9523     w.alert(note+' : ' + v + '\n'
9524         + '-- URL: ' + u + '\n'
9525         + '-- Time: ' + t + ' (' + DateLong0(t*1000) + ')');
9526 };
9527
9528 function html_save(){
9529     u = document.URL;
9530     t = new Date().getTime() / 1000;
9531     v = '<'+span id="+htVersionTag+" data-url="+u+" data-time="+t+">";
9532     v += '<'+/span>\n';
9533     h += v + htRoot.outerHTML;
9534     localStorage.setItem(snappedHTML,h);
9535     showVersion("Saved",window,v,u,t);
9536 }
9537 function html_load(){
9538     h = localStorage.getItem(snappedHTML);
9539     if( h == null ){
9540         alert('No snapshot taken yet');
9541         return;
9542     }
9543     w = window.open('','','');
9544     d = w.document;
9545     d.write(h);
9546     w.focus();
9547     html_ver1("Loaded",w,d);
9548 }

```

```

9549 function html_ver1(note,w,d){
9550   if( (v = d.getElementById(htVersionTag)) != null ){
9551     h = v.outerHTML;
9552     u = v.getAttribute('data-url');
9553     t = v.getAttribute('data-time');
9554   }else{
9555     h = 'No version info. in the page';
9556     u = '';
9557     t = 0;
9558   }
9559   showVersion(note,w,v,u,t);
9560 }
9561 function html_ver0(){
9562   html_ver1("Version",window,document);
9563 }
9564 </script>
9565 <!-- LiveHTML -->
9566 </span>
9567 </details>
9568 */
9569 /*
9570 */
9571 <details><summary>Event sharing</summary>
9572 <span id="EventSharingCodeSpan">
9573
9574 <!-- ----- Event sharing // 2020-0925 SatoxITS { -->
9575
9576 <div id="iftestTemplate" class="iftest" hidden="">
9577 <style> .iftestbody{ color:#f22;font-family:Georgia;font-size:10pt;} </style>
9578 <span id="frameBody" class="iftestbody" onclick="frameClick()"><script>
9579   function docadd(txt){
9580     document.body.append(txt);
9581     window.scrollTo(0,100000);
9582   }
9583   function frameClick(){
9584     xy = '(x='+event.x + ' y='+event.y+')';
9585     //docadd('Got Click on #' +event.target.id+ ' +xy+ '\n');
9586     docadd('Got Click on #' +Fid.value+ ' +xy+ '\n');
9587     window.scrollTo(0,100000);
9588     window.parent.postMessage('OnClick: '+xy, '*');
9589   }
9590   function frameMousemove(){
9591     if( false ){
9592       document.body.append('Mousemove on #' +event.target.id+ ' '
9593         + 'x='+event.x + ' y='+event.y + '\n');
9594       peerWin = window.frames.iframe1;
9595       document.body.append('Send to peer #' +peerWin+ ' ' + '\n');
9596       window.scrollTo(0,100000);
9597       peerWin.postMessage('Hi!', '*');
9598     }
9599   }
9600   function frameKeydown(){
9601     msg = 'Got Keydown: #' +Fid.value+ ' (' +event.code+ ' ';
9602     docadd(msg+ '\n');
9603     window.parent.postMessage(msg, '*');
9604   }
9605   function frameOnMessage(){
9606     docadd('Message ' + event.data + '\n');
9607     window.scrollTo(0,100000);
9608   }
9609   if( document.getElementById('Fid') ){
9610     frameBody.id = Fid.value;
9611     h = '';
9612     h += '<'+style>*<';
9613     h += 'font-size:10pt;white-space:pre-wrap;';
9614     h += 'font-family:Courier New;';
9615     h += '<'+style>';
9616     h += 'I am '+Fid.value+ '\n';
9617     document.write(h);
9618     window.addEventListener('click',frameClick);
9619     window.addEventListener('keydown',frameKeydown);
9620     window.addEventListener('message',frameOnMessage);
9621     window.addEventListener('mousemove',frameMousemove);
9622     window.parent.postMessage('Hi parent, I am '+Fid.value, '*');
9623   }
9624 </script></span></div>
9625
9626 <style>.iframeTest{margin:3px;resize:both;width:370px;height:120px}</style>
9627 <h2>Inter-window communicaiton</h2>
9628 <note>
9629 frame0 >>> frame1 and frame2<br>
9630 frame1 >>> frame0 and frame2<br>
9631 frame2 >>> frame0 and frame1<br>
9632 </note>
9633 <div id="iframe-test">
9634 <pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
9635 <iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
9636 <iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
9637 <iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
9638 </div>
9639
9640 <script id="if0-test-script">
9641   setupFrames0();
9642   setupFrames12();
9643
9644   function setFrameSrcdoc(dst,src){
9645     if( true ){
9646       dst.contentWindow.document.write(src);
9647       // this makes browser wait close, and crash if accumulated !?
9648       // so it should be closed after write
9649       dst.contentWindow.document.close();
9650     }else{
9651       // to be erased before source dump
9652       // but should be set for live snapshot
9653       dst.srcdoc = src;
9654     }
9655   }
9656   function setupFrames0(){
9657     ibody = iframe0.contentWindow.document.body;
9658     iframe0.style.width = "755px";
9659     //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
9660     window.addEventListener('message',messageFromChild);
9661
9662     if0 = '';
9663     if0 += '<'+pre style="font-family:Courier New;">';
9664     if0 += '<input id="Fid" value="iframe0">';
9665     if0 += iftestTemplate.innerHTML;
9666     setFrameSrcdoc(iframe0,if0);
9667
9668     function clickOnChild(){
9669       console.log('clickOn #' +this.id);
9670     }
9671     function moveOnChild(){
9672       console.log('moveOn #' +this.id);

```

```

9673     }
9674     iframe0.contentWindow.document.body.style.setProperty('white-space', 'pre');
9675     iframe0.contentWindow.document.body.style.setProperty('font-size', '9pt');
9676   }
9677   function setupFrames12(){
9678     if1 = '<input id="Fid" value="iframe1">';
9679     if1 += iftestTemplate.innerHTML;
9680     setFrameSrcdoc(iframe1,if1);
9681     //iframe1.name = 'iframe1'; // this seems break contentWindow
9682
9683     if2 = '<input id="Fid" value="iframe2">';
9684     if2 += iftestTemplate.innerHTML;
9685     setFrameSrcdoc(iframe2,if2);
9686
9687     iframe1.addEventListener('message',messageFromChild);
9688     //iframe1.addEventListener('mouseover',moveOnChild);
9689     iframe2.addEventListener('message',messageFromChild);
9690     //iframe2.addEventListener('mouseover',moveOnChild);
9691     iframe1.contentWindow.postMessage('[parent0] Hi iframe1 -- from parent.','*');
9692     //iframe1.contentWindow.postMessage('Your peer is '+iframe2.contentWindow,'*');
9693     iframe2.contentWindow.postMessage('[parent0] Hi iframe2 -- from parent.','*');
9694     //iframe2.contentWindow.postMessage('Your peer is '+iframe1.contentWindow,'*');
9695   }
9696   function messageFromChild(){
9697     from = null;
9698     forw = null;
9699     if( event.source == iframe0.contentWindow ){
9700       from = 'iframe0'
9701       forw = 'iframe12';
9702     }else
9703     if( event.source == iframe1.contentWindow ){
9704       from = 'iframe1'
9705       forw = 'iframe2';
9706     }else
9707     if( event.source == iframe2.contentWindow ){
9708       from = 'iframe2'
9709       forw = 'iframe1';
9710     }else
9711     {
9712       iframeHost.innerHTML += 'Message [unknown] '
9713       + ' orig=' + event.origin
9714       + ' data=' + event.data
9715       //+ ' from=' + event.source
9716       ;
9717     }
9718     msglog1 = from + event.data + ' -- '
9719     + ' from=' + event.source
9720     + ' orig=' + event.origin
9721     + ' name=' + event.source.name
9722     //+ ' port=' + event.ports
9723     //+ ' evid=' + event.lastEventId
9724     + '\n'
9725     ;
9726     if( true ){
9727       if( forw == 'iframe1' || forw == 'iframe12' ){
9728         iframe1.contentWindow.postMessage(from+event.data);
9729       }
9730       if( forw == 'iframe2' || forw == 'iframe12' ){
9731         iframe2.contentWindow.postMessage(from+event.data);
9732       }
9733     }
9734     txtadd0(msglog1);
9735
9736     function txtadd0(txt){
9737       iframe0.contentWindow.document.body.append(txt);
9738       iframe0.contentWindow.scrollTo(0,100000);
9739     }
9740
9741     function es_ShowSelf(){
9742       iframe1.setAttribute('src',document.URL);
9743       iframe2.setAttribute('src',document.URL);
9744     }
9745   </script>
9746
9747   <input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
9748   <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
9749   <span id="EventSharingCodeview"></span>
9750   <script id="EventSharingScript">
9751   function es_showHtmlCode(){
9752     showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
9753   }
9754   DestroyEventSharingCodeview = function(){
9755     //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
9756     EventSharingCodeview.innerHTML = "";
9757     iframe0.style = "";
9758     //iframe0.srcdoc = "erased";
9759     //iframe1.srcdoc = "erased";
9760     //iframe2.srcdoc = "erased";
9761   }
9762   </script>
9763   <!-- EventSharing -->
9764   </span>
9765   </details>
9766   */
9767
9768   /*
9769   <!-- ----- "GShell Inside" Notification { -->
9770   <script id="script-gshell-inside">
9771   var notices = 0;
9772   function noticeGShellInside(){
9773     ver = '';
9774     if( ver = document.getElementById('GshVersion') ){
9775       ver = ver.innerHTML;
9776     }
9777     console.log('GJShell Inside (^-^)'//'+ver);
9778     notices += 1;
9779     if( 2 <= notices ){
9780       document.removeEventListener('mousemove',noticeGShellInside);
9781     }
9782   }
9783   document.addEventListener('mousemove',noticeGShellInside);
9784   noticeGShellInside();
9785
9786   const FooterName = 'GshFooter'
9787   function DestroyFooter(){
9788     if( (footer = document.getElementById(FooterName)) != null ){
9789       //footer.parentNode.removeChild(footer);
9790       empty = document.createElement('div');
9791       empty.id = 'GshFooter0';
9792       footer.parentNode.replaceChild(empty,footer);
9793     }
9794   }
9795
9796   footer = document.createElement('div');

```

```

9797 footer.id = FooterName;
9798 footer.style.backgroundImage = "url("+ITSmoreQR+)";
9799 //GshFooter0.parentNode.appendChild(footer);
9800 GshFooter0.parentNode.replaceChild(footer,GshFooter0);
9801 </script>
9802 <!-- } -->
9803
9804 <!--
9805     border:20px inset #888;
9806 -->
9807
9808 //<span id="WirtualDesktopCodeSpan">
9809 /*
9810 <details><summary>Wirtual Desktop</summary>
9811 <!-- ----- Web Wirtual Desktop // 2020-0927 SatoxITS { -->
9812 <style>
9813 .WirtualSpace {
9814     z-index:0;
9815     xwidth:1280px !important; xheight:720px !important;
9816     width:5120px; height:2880px;
9817     border-width:0px;
9818     xxxbackground-color:rgba(32,32,160,0.8);
9819     xxxbackground-image:url("WD-WallPaper03.png");
9820     xxxbackground-size:100% 100%;
9821     color:#22a;xfont-family:Georgia;font-size:10pt;
9822     xxoverflow:scroll;
9823 }
9824 .WirtualGrid {
9825     z-index:0;
9826     position:absolute;
9827     width:800px; height:500px;
9828     border:1px inset #fff;
9829     color:rgba(192,255,192,0.8);
9830     font-family:Georgia, Courier New;
9831     text-align:right;
9832     vertical-align:middle;
9833     font-size:200px;
9834     text-shadow:4px 4px #ccf;
9835 }
9836 .WD_GridScroll {
9837     z-index:100000;
9838     background-color:rgba(200,200,200,0.1);
9839 }
9840 .WirtualDesktop {
9841     z-index:0;
9842     position:relative;
9843     resize:both !important;
9844     overflow:scroll;
9845     display:block;
9846     min-width:120px !important; min-height:60px !important;
9847     width:800px;
9848     height:500px;
9849     border:10px inset #228;
9850     border-width:30px; border-radius:20px;
9851     background-image:url("WD-WallPaper03.png");
9852     background-size:100% 100%;
9853     color:#22a;font-family:Georgia;font-size:10pt;
9854 }
9855 .comment {
9856     // overflow=scroll seems to bound childrens' view in the element span
9857     // specifying overflow seems fix the position of the element
9858 }
9859 .WirtualBrowserSpan {
9860     z-index:10;
9861     xxxborder:0.5px dashed #fff !important;
9862     border-color:rgba(255,255,255,0.5) !important;
9863     position:relative;
9864     left:100px;
9865     top:100px;
9866     display:block;
9867     resize:both !important;
9868     width:540px;
9869     height:320px;
9870     min-width:40px !important; min-height:20px !important;
9871     max-width:5120px !important; max-height:2880px !important;
9872     background-color:rgba(255,200,255,0.1);
9873     xoverflow:scroll;
9874 }
9875 .xWirtualBrowserLocationBar:focus {
9876     color:#f00;
9877     background-color:rgba(255,128,128,0.2);
9878 }
9879 .xWirtualBrowserLocationBar:active {
9880     color:#f00;
9881     background-color:rgba(128,255,128,0.2);
9882 }
9883 a.WirtualBrowserLocation {
9884     color:#ccc !important;
9885     text-decoration:none !important;
9886 }
9887 a.WirtualBrowserLocation:hover {
9888     color:#fff !important;
9889     text-decoration:underline;
9890 }
9891 .WirtualBrowserLocationBar {
9892     position:absolute;
9893     z-index:100000;
9894     display:block;
9895     width:400px;
9896     height:20px;
9897     padding-left:2px;
9898     line-height:1.1;
9899     vertical-align:middle;
9900     font-size:14px;
9901     color:#fff;
9902     background-color:rgba(128,128,128,0.2);
9903     font-family:Georgia;
9904 }
9905 .WirtualBrowserCommandBar {
9906     position:absolute;
9907     z-index:200000;
9908     xxxdisplay:inline;
9909     display:block;
9910     width:60px;
9911     height:20px;
9912     line-height:1.1;
9913     vertical-align:middle;
9914     font-size:14px;
9915     color:#fe4;
9916     background-color:rgba(128,128,128,0.1);
9917     font-family:Georgia;
9918     text-align:left;
9919     left:404px;
9920 }

```



```

9921 .VirtualBrowserFrame {
9922   xxposition:relative;
9923   position:absolute;
9924   xxdisplay:inline;
9925   display:block;
9926   z-index:10;
9927   resize:both !important;
9928   width:480px; height:240px;
9929   min-width:60px; min-height:30px;
9930   max-width:5120px; max-height:2880px;
9931   border-radius:6px;
9932   background-color:rgba(255,255,255,0.9);
9933   border-top:20px solid;
9934   border-right:4px solid;
9935   border-bottom:10px solid;
9936 }
9937 .WinFavicon {
9938   width:16px;
9939   height:16px;
9940   margin:1px;
9941   margin-right:3px;
9942   vertical-align:middle;
9943   background-color:rgba(255,255,255,1.0);
9944 }
9945 .VirtualDesktopMenuBar {
9946   xposition:absolute;
9947   color:#fff;
9948   font-size:7pt;
9949   text-align:right;
9950   padding-right:4px;
9951   background-color:rgba(128,128,128,0.7);
9952 }
9953 .VirtualDesktopCalender {
9954   color:#fff;
9955   font-size:22pt;
9956   text-align:right;
9957   padding-right:4px;
9958   xbackground-color:rgba(255,255,255,0.2);
9959 }
9960 .xxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9961   display:inline !important; font-size:10pt !important; padding:1px !important;
9962 }
9963 .WD_Config {
9964   display:inline !important;
9965   padding:2px !important;
9966   font-size:10pt !important;
9967   width:60pt !important;
9968   height:12pt !important;
9969   line-height:1.0pt !important;
9970   height:15pt !important;
9971 }
9972 .WD_Button {
9973   display:inline !important;
9974   padding:2px !important;
9975   color:#fff !important;
9976   background-color:#228 !important;
9977   font-size:10pt !important;
9978   width:60pt !important;
9979   height:12pt !important;
9980   line-height:1.0pt !important;
9981   height:16pt !important;
9982   border:2px inset #44a !important;
9983 }
9984 .WD_Href {
9985   display:inline !important;
9986   padding:2px !important;
9987   font-size:9pt !important;
9988   width:120pt !important;
9989   height:12pt !important;
9990   line-height:1.0pt !important;
9991   height:15pt !important;
9992 }
9993 }
9994 .LiveHtmlCodeviewText {
9995   font-size:10pt;
9996   font-family:Courier New;
9997   xwhite-space:pre;
9998 }
9999 }
10000 .WD_Panel {
10001   x-index:100 !important;
10002   color:#000 !important;
10003   margin-left:25px !important;
10004   width:800px !important;
10005   padding:4px !important;
10006   border:1px solid #888 !important;
10007   border-radius:6px !important;
10008   background-color:rgba(220,220,220,0.9) !important;
10009   font-size:9pt;
10010   font-family:Courier New;
10011 }
10012 .WD_Help {
10013   font-size:10pt !important;
10014   font-family:Courier New;
10015   line-height:1.2 !important;
10016   color:#000 !important;
10017   width:100% !important;
10018   background-color:rgba(240,240,255,0.8) !important;
10019 }
10020 }
10021 .WB_Zoom {
10022 }
10023 </style>
10024 <h2>CosmoScreen 0.0.8</h2>
10025 <menu>
10026 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br><br>
10027 g ... grid on/off<br>
10028 i ... zoom in<br>
10029 o ... zoom out<br>
10030 s ... save current scope<br>
10031 r ... restore saved scope<br>
10032 </span>
10033 </menu>
10034 <div class="WD_Panel" draggable="true">
10035 <p><!-- should be on the frame of the WD -->
10036 Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
10037 < <input id="WD_Width_1" class="WD_Config" type="text">
10038 x <input id="WD_Height_1" class="WD_Config" type="text">
10039 wall-paper: <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaper03.png</a>
10040 </p>
10041 <p>
10042 Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
10043 < <input id="WS_1_Width" class="WD_Config" type="text">
10044 x <input id="WS_1_Height" class="WD_Config" type="text">

```

```

10045</p>
10046<p>
10047Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
10048 < <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
10049 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
10050 < <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
10051 < <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
10052</p>
10053<p>
10054Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
10055 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
10056 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
10057shift+wheel for horizontal scroll
10058</p>
10059<p>
10060Scopexx <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
10061 < <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
10062 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
10063 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
10064</p>
10065<p>
10066Scopeyy <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
10067 < <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
10068 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
10069 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
10070</p>
10071<p>
10072Scopezz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
10073 < <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
10074 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
10075 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
10076</p>
10077<p>
10078Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
10079</p>
10080<p>
10081Overflow <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
10082"scroll" imprisons windows inside the display
10083</p>
10084</div>
10085
10086<div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" contenteditable="true" style="">
10087<div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
10088<i>CosmosScreen 0.0.8</i><span id="VirtualDesktop_1_Clock"></span>
10089</div>
10090<div id="VirtualDesktop_1_Calender" class="VirtualDesktopCalender" >00:00</div>
10091<div align=right><h1>VirtualSpace 1.</h1></div>
10092<div id="VirtualDesktop_1_Content" class="VirtualSpace">
10093
10094<div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10095<div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar"></div>
10096<span id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar">Reload</span>
10097<iframe id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""></iframe>
10098</div>
10099
10100<div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10101<div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
10102<span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar">Reload</span>
10103<iframe id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""></iframe>
10104</div>
10105
10106<div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
10107<div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar"></div>
10108<span id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar">Reload</span>
10109<iframe id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""></iframe>
10110</div>
10111
10112<div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
10113</div>
10114</div>
10115
10116<input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">
10117<span id="VirtualDesktopCodeview"></span>
10118<script id="VirtualDesktopScript">
10119function vd_showHtmlCode(){
10120 codespan = document.getElementById('VirtualDesktopCodeSpan');
10121 showHtmlCode(VirtualDesktopCodeview, codespan);
10122 VirtualDesktopCodeview.setAttribute('class', 'LiveHtmlCodeviewText');
10123}
10124DestroyEventSharingCodeview = function(){
10125 VirtualDesktopCodeview.innerHTML = "";
10126}
10127
10128function wdlog(log){
10129 if( GJ_Channel != null ){
10130 GJ_SendMessage('WD '+log);
10131 }
10132 console.log(log);
10133}
10134var topMostWin = 10000;
10135function onEnterWin(e){
10136 t = e.target;
10137 oindex = t.style.zIndex;
10138 //if( oindex == '' ) oindex = 0;
10139 //t.saved_zIndex = oindex;
10140 //t.style.zIndex = 10000;
10141 topMostWin += 1;
10142 t.style.zIndex = topMostWin;
10143 nindex = t.style.zIndex;
10144 wdlog('Enter '+t+' #' +t.id+' ('+oindex+'->'+nindex+'');
10145 e.stopPropagation();
10146 e.preventDefault();
10147}
10148function onClickWin(e){ // can detect click on the thick border? t = e.target;
10149 oindex = t.style.zIndex;
10150 topMostWin += 1;
10151 t.style.zIndex = topMostWin;
10152 nindex = t.style.zIndex;
10153 wdlog('Click '+t+' #' +t.id+' ('+oindex+'->'+nindex+'');
10154 //e.stopPropagation();
10155 //e.preventDefault();
10156}
10157function onLeaveWin(e){
10158 t = e.target;
10159 //oindex = t.style.zIndex;
10160 //nindex = t.saved_zIndex;
10161 //t.style.zIndex = nindex;
10162 //wdlog('Leave '+e.target+' #' +e.target.id+' ('+oindex+'->'+nindex+'');
10163 e.stopPropagation();
10164 e.preventDefault();
10165}
10166
10167var WinDragstartX; // event
10168var WinDragstartY;

```

```

10169var WinDragstartTX; // target
10170var WinDragstartTY;
10171
10172function onWinDragstart(e){
10173    WinDragstartX = e.x;
10174    WinDragstartY = e.y;
10175
10176    t = e.target;
10177
10178    //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
10179    //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
10180    if( t.style.left == '' ){
10181        WinDragstartTX = x0 = 0;
10182        t.style.left = '0px';
10183    }else{
10184        //WinDragstartTX = x0 = Number(t.style.left);
10185        WinDragstartTX = x0 = parseInt(t.style.left);
10186    }
10187    if( t.style.top == '' ){
10188        WinDragstartTY = y0 = 0;
10189        t.style.top = '0px';
10190    }else{
10191        //WinDragstartTY = y0 = Number(t.style.top);
10192        WinDragstartTY = y0 = parseInt(t.style.top);
10193    }
10194    if( true ){ // to be undo
10195        t.wasAtX = WinDragstartTX;
10196        t.wasAtY = WinDragstartTY;
10197    }
10198    wdlog('DragSTA #' + t.id
10199        + ' event(' + e.x + ',' + e.y + ')'
10200        + ' position=' + t.style.position
10201        + ' style left,top(' + t.style.left + ',' + t.style.top + ')'
10202    );
10203    e.stopPropagation();
10204    //e.preventDefault();
10205    return true;
10206}
10207function onWinDragEvent(wh,e,set,dolog){
10208    t = e.target;
10209    dx = e.x - WinDragstartX;
10210    dy = e.y - WinDragstartY;
10211    nx = WinDragstartTX + dx;
10212    ny = WinDragstartTY + dy;
10213    log = 'Drag'+wh+' #' + t.id
10214        + ' event0(' + WinDragstartX + ',' + WinDragstartY + ')'
10215        + ' event(' + e.x + ',' + e.y + ')'
10216        + ' diff(' + dx + ',' + dy + ')'
10217        + ' (' + nx + ',' + ny + ')'
10218        + ' (' + t.style.left + ',' + t.style.top + ')'
10219        + ' wasAt(' + t.wasAtX + ',' + t.wasAtY + ')'
10220    ;
10221    if( e.x != 0 || e.y != 0 ){
10222        if( set == true ){
10223            //t.style.x = nx + 'px'; // not effective
10224            //t.style.y = ny + 'px'; // not effective
10225            t.style.left = nx + 'px';
10226            t.style.top = ny + 'px';
10227            log += ' Set';
10228        }else{
10229            log += ' NotSet';
10230            if( !dolog ){
10231                log = '';
10232            }
10233        }
10234    }else{
10235        log += ' What?'; // the type is event start?
10236        if( !dolog ){
10237            log = '';
10238        }
10239    }
10240    if( and(dolog, log != '' ) ){
10241        wdlog(log);
10242    }
10243    if( true ){
10244        // should be propargeted to parent in FireFox ?
10245        e.stopPropagation();
10246    }
10247    e.preventDefault();
10248    return false;
10249}
10250function onWinDrag(e){
10251    return onWinDragEvent('Ing',e,true,false);
10252}
10253function onWinDragend(e){
10254    return onWinDragEvent('End',e,false,true);
10255}
10256function onWinDragexit(e){
10257    return onWinDragEvent('Exit',e,false,true);
10258}
10259function onWinDragover(e){
10260    return onWinDragEvent('Over',e,false,true);
10261}
10262function onWinDragenter(e){
10263    return onWinDragEvent('Enter',e,false,true);
10264}
10265function onWinDragleave(e){
10266    return onWinDragEvent('Leave',e,false,true);
10267}
10268function onWinDragdrop(e){
10269    return onWinDragEvent('Drop',e,false,true);
10270}
10271function onFaviconChange(e){
10272    wdlog('--Favicon #' + e.target.id + ' href=' + e.details.href);
10273}
10274var savedSuppressGJShell = false;
10275function stopGShell(e){
10276    //wdlog('enter Gsh STOP');
10277    savedSuppressGJShell = SuppressGJShell;
10278    SuppressGJShell = true;
10279    e.stopPropagation();
10280    e.preventDefault();
10281}
10282function contGShell(e){
10283    //wdlog('leave Gsh STOP');
10284    SuppressGJShell = savedSuppressGJShell;
10285    e.stopPropagation();
10286    e.preventDefault();
10287}
10288
10289function WD_onKeydown(e){
10290    keycode = e.code;
10291    console.log('Keydown #' + e.target.id + ' ' + keycode);
10292    if( keycode == 'KeyG' ){

```

```

10293     WD_setGrid1(WD_Grid1);
10294 }else
10295 if( keycode == 'KeyI' ){
10296     WD_doZoomIN();
10297 }else
10298 if( keycode == 'KeyO' ){
10299     WD_doZoomOUT();
10300 }else
10301 if( keycode == 'KeyR' ){
10302     WD_RestoreScope(null);
10303 }else
10304 if( keycode == 'KeyS' ){
10305     WD_SaveScope(null);
10306 }
10307 e.stopPropagation();
10308 e.preventDefault();
10309 }
10310 function WD_onKeyUp(e){
10311     e.stopPropagation();
10312     e.preventDefault();
10313 }
10314 VirtualDesktop_1.addEventListener('keydown', e => { WD_onKeyDown(e); });
10315 VirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onKeyDown(e); });
10316 WD_Help_1.addEventListener('keydown', e => { WD_onKeyDown(e); });
10317 WD_Help_1.addEventListener('keyup', e => { WD_onKeyUp(e); });
10318 }
10319 function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
10320     function WirtualBrowserCommand(e,s,l,cmd,f){
10321         command = cmd.innerHTML
10322         if( command == "Reload" ){
10323             href_id = e.target.href_id;
10324             d = document.getElementById(href_id);
10325             wlog('href_tag=#'+href_id+'\n elem=#'+href_id+'\n href='+d);
10326             url = d.innerHTML;
10327             wlog('---- Load href_tag=#'+href_id+'\n elem=#'+href_id+'\n href='+d
10328                 +'\n url='+url);
10329             wlog('---- Load target #' + f.id + ' with url=' + url;
10330             f.src = url;
10331         }else{
10332             alert('unknown command'+command+' '+e.target.id+', '+l.id+', '+f.id);
10333         }
10334     }
10335     function onKeyDown(e){
10336         if( e.code == 'Enter' ){
10337             e.stopPropagation();
10338             e.preventDefault();
10339         }
10340     }
10341     function onKeyUp(e){
10342         if( e.code == 'Enter' ){
10343             e.stopPropagation();
10344             e.preventDefault();
10345             // should reload immediately ?
10346         }
10347     }
10348 }
10349 if( false ){
10350     wlog('start settle WirtualBrowser url='+u +'\n'
10351         + 'id=' + s.id + '\n'
10352         + 'width=' + s.style.width + '\n'
10353         + 'height=' + s.style.height
10354     );
10355 }
10356 // very iportant for WordPress ??
10357 s.style.width = f.style.width = 501; // for WordPress ...??
10358 s.style.height = f.style.height = 271; // for WordPress ...??
10359 if( false ){
10360     wlog('midway settle WirtualBrowser url='+u +'\n'
10361         + 'id=' + s.id + '\n'
10362         + 'width=' + s.style.width + '\n'
10363         + 'height=' + s.style.height
10364     );
10365 }
10366 s.width = 502; // for WordPress ...??
10367 s.height = 272; // for WordPress ...??
10368 if( false ){
10369     wlog('midway-2 settle WirtualBrowser url='+u +'\n'
10370         + 'id=' + s.id + '\n'
10371         + 'span-width=' + s.width + '\n'
10372         + 'span-height=' + s.height
10373     );
10374 }
10375 }
10376 s.style.width = w + 'px';
10377 s.style.height = h + 'px';
10378 f.style.width = w + 'px';
10379 f.style.height = h + 'px';
10380 //f.style.setProperty('-webkit-transform', 'scale('+scale+')');
10381 f.style.setProperty('transform', 'scale('+scale+')');
10382 }
10383 //wlog('-x1-- u='+u+ ' width s='+s.style.width+',f='+f.style.width);
10384 //wlog('-x2-- u='+u+ ' width s='+s.style.width+',f='+f.style.width);
10385 s.setAttribute('draggable','true');
10386 f.setAttribute('draggable','false'); // why necessary?
10387 l.setAttribute('draggable','false'); // why necessary?
10388 cmd.setAttribute('draggable','false'); // why necessary?
10389 s.addEventListener('dragstart', e => { onWinDragstart(e); });
10390 s.addEventListener('drag', e => { onWinDrag(e); });
10391 s.addEventListener('exit', e => { onWinDragexit(e); });
10392 s.addEventListener('dragend', e => { onWinDragend(e); });
10393 s.addEventListener('dragexit', e => { onWinDragexit(e); });
10394 s.addEventListener('dragenter', e => { onWinDragenter(e); });
10395 s.addEventListener('dragover', e => { onWinDragover(e); });
10396 s.addEventListener('dragleave', e => { onWinDragleave(e); });
10397 s.addEventListener('drop', e => { onWinDragdrop(e); });
10398 }
10399 s.addEventListener('mouseenter',e => { onEnterWin(e); });
10400 s.addEventListener('mouseleave',e => { onLeaveWin(e); });
10401 }
10402 if( false ){
10403     s.style.position = "absolute";
10404     s.style.x = x+'px';
10405     s.style.left = x+'px';
10406     s.style.y = y+'px';
10407     s.style.top = y+'px';
10408 }else{
10409     s.style.setProperty('position','absolute','important');
10410     s.style.setProperty('x',x+'px','important');
10411     s.style.setProperty('left',x+'px','important');
10412     s.style.setProperty('y',y+'px','important');
10413     s.style.setProperty('top',y+'px','important');
10414 }
10415 }
10416 favicon = './favicon.ico';

```

```

10417 uv1 = u.split('/:');
10418 if( 2 <= uv1.length ){
10419     uv2 = uv1[1].split('/');
10420     if( 2 <= uv2.length ){
10421         if( uv1[0] == 'file' ){
10422             //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/');
10423             // + '/favicon.ico';
10424             favicon = './favicon.ico';
10425         }else{
10426             favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
10427         }
10428     }
10429 }
10430 //wlog("----- favicon-url="+favicon);
10431 href_id = l.id + '_href';
10432 l.innerHTML = ''+u+'</a>';
10434 //l.addEventListener('click', e => { onClickWin(e); });
10435 l.addEventListener('mouseenter',e => { stopGShell(e); });
10436 l.addEventListener('mouseleave',e => { contGShell(e); });
10437 l.addEventListener('keydown', e => { onKeyDown(e); });
10438 l.addEventListener('keyup', e => { onKeyUp(e); });
10439
10440 cmd.href_id = href_id;
10441 wlog('(0)cmd=#'+cmd.id);
10442 wlog('(1)href_id=#'+href_id);
10443 wlog('(2)href_id=#'+cmd.href_id);
10444 cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
10445
10446 f.style.borderColor = c;
10447 f.src = u;
10448 //f.addEventListener('mouseenter',e => { onEnterWin(e); });
10449 //f.addEventListener('mouseleave',e => { onLeaveWin(e); });
10450
10451 //s.addEventListener('click', e => { onClickWin(e); });
10452 //f.addEventListener('click', e => { wlog('click wbl'); });
10453 f.addEventListener('mozbrowsericonchange',onFaviconChange);
10454
10455 wlog('done settle VirtualBrowser url='+u +'\n'
10456 + 'id=' + s.id + ' '
10457 + 'width=' + s.style.width + ' '
10458 + 'height=' + s.style.height + ' '
10459 + 'cmd=' + cmd.id
10460 );
10461}
10462
10463dt = VirtualDesktop 1;
10464dt.style.width = "800px";
10465dt.style.height = "500px";
10466dt.addEventListener('dragstart',e => { onWinDragstart(e); });
10467dt.addEventListener('drag', e => { onWinDrag(e); });
10468dt.addEventListener('exit', e => { onWinDragexit(e); });
10469
10470function GRonClick(){
10471    WD_SaveScope(null); // should be push
10472    t = event.target;
10473    x = t.getAttribute('data-leftx');
10474    y = t.getAttribute('data-topy');
10475    zoom = WD_Zoom_1_XY.value;
10476    x *= zoom;
10477    y *= zoom;
10478    WD_doScrollXY(event,x,y);
10479    //alert('scroll #' + t.id + ' x=' + x + ', y=' + y);
10480}
10481function WD_setGrid(e){
10482    t = e.target;
10483    WD_setGrid1(t);
10484}
10485function WD_setGrid1(t){
10486    //ds = VirtualDesktop 1_Content; // should be VirtualSpace 1
10487    ds = VirtualDesktop 1_GridPlane;
10488    if( t.value == 'GridOn' ){
10489        for( col = 0; col < 16; col++ ){
10490            for( row = 0; row < 16; row++ ){
10491                gl = document.createElement('span');
10492                gl.setAttribute('class','VirtualGrid');
10493                leftx = col * 800;
10494                topy = row * 500;
10495                gid = col + '.' + row
10496                label = '<'+span '
10497                    + 'id="'+gid+'" '+class="WD_GridScroll" '
10498                    + 'contenteditable="false" onclick="GRonClick()" '
10499                    + 'data-leftx=' + leftx + ' ' + 'data-topy=' + topy + ' '
10500                    + '>'
10501                + gid + '<'+/span>';
10502                console.log('grid '+label);
10503                gl.innerHTML = label;
10504                gl.position = 'relative';
10505                gl.leftx = leftx;
10506                gl.topy = topy;
10507                gl.style.left = gl.leftx + 'px';
10508                gl.style.top = gl.topy + 'px';
10509                if( col % 2 == row % 2 ){
10510                    gl.style.backgroundColor = 'rgba(255,255,255,0.3)';
10511                }
10512                ds.appendChild(gl);
10513            }
10514        }
10515        t.value = 'GridOff';
10516    }else{
10517        ds.innerHTML = '';
10518        t.value = 'GridOn';
10519    }
10520}
10521WD_Grid 1.addEventListener('click', e => { WD_setGrid(e); });
10522
10523var sav_scrollLeft;
10524var sav_scrollTop;
10525var sav_nscale;
10526function WD_SaveScope(e){
10527    sav_scrollLeft = WD_Left 1.value;
10528    sav_scrollTop = WD_Top 1.value;
10529    sav_nscale = WD_Zoom 1_XY.value;
10530    //console.log('saved zoom='+sav_oscale+','+sav_nscale);
10531}
10532function WD_RestoreScope(e){
10533    WD_Zoom 1_XY.value = sav_nscale;
10534    WD_doZoom();
10535
10536    WD_Left 1.value = sav_scrollLeft;
10537    WD_Top 1.value = sav_scrollTop;
10538    WD_doScroll(null);
10539}
10540WD_Zoom 1_Save.addEventListener('click', e => { WD_SaveScope(e); });

```

```

10541WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
10542
10543function ignoreEvent(e){
10544    e.stopPropagation();
10545    //e.preventDefault();
10546}
10547WD_Width_1.value = dt.style.width;
10548WD_Width_1.addEventListener('keydown',ignoreEvent);
10549WD_Width_1.addEventListener('keyup',ignoreEvent);
10550WD_Height_1.value = dt.style.height;
10551WD_Height_1.addEventListener('keydown',ignoreEvent);
10552WD_Height_1.addEventListener('keyup',ignoreEvent);
10553
10554function zoomMag(){
10555    return WD_Zoom_1_MAG.value;
10556}
10557WD_Zoom_1_MAG.addEventListener('keydown',ignoreEvent);
10558WD_Zoom_1_MAG.addEventListener('keyup',ignoreEvent);
10559
10560function escale1(e,oscale,nscale){
10561    e.style.setProperty('transform','scale('+nscale+')');
10562    rscale = oscale / nscale;
10563    w = parseInt(e.style.width);
10564    h = parseInt(e.style.height);
10565    w = w * rscale; //(oscale/nscale);
10566    h = h * rscale; //(oscale/nscale);
10567    e.style.width = w + 'px';
10568    e.style.height = h + 'px';
10569}
10570function scaleWD(ds,oscale,nscale){
10571    if( true ){
10572        escale1(WirtualBrowser_1,oscale,nscale);
10573        escale1(WirtualBrowser_1_Location,oscale,nscale);
10574        escale1(WirtualBrowser_1_Command,oscale,nscale);
10575        escale1(WirtualBrowser_1_Frame,oscale,nscale);
10576
10577        escale1(WirtualBrowser_2,oscale,nscale);
10578        escale1(WirtualBrowser_2_Location,oscale,nscale);
10579        escale1(WirtualBrowser_2_Command,oscale,nscale);
10580        escale1(WirtualBrowser_2_Frame,oscale,nscale);
10581
10582        escale1(WirtualBrowser_3,oscale,nscale);
10583        escale1(WirtualBrowser_3_Location,oscale,nscale);
10584        escale1(WirtualBrowser_3_Command,oscale,nscale);
10585        escale1(WirtualBrowser_3_Frame,oscale,nscale);
10586    }
10587}
10588function WD_doZoom(){
10589    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10590    oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
10591    nscale = WD_Zoom_1_XY.value;
10592    ds.style.zoom = nscale;
10593    WD_Zoom_1_XY.value = ds.style.zoom;
10594    WD_Zoom_1_XY.ovalue = ds.style.zoom;
10595    scaleWD(ds,oscale,nscale);
10596}
10597WD_Zoom_1.addEventListener('click',WD_doZoom);
10598WD_Zoom_1_XY.addEventListener('keydown',ignoreEvent);
10599WD_Zoom_1_XY.addEventListener('keyup',ignoreEvent);
10600
10601function WD_doZoomOUT(){
10602    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10603    oscale = WD_Zoom_1_XY.value;
10604    if( oscale == 0 || oscale == '' ){
10605        oscale = 1;
10606    }
10607    nscale = oscale / zoomMag();
10608    ds.style.zoom = nscale;
10609    WD_Zoom_1_XY.value = ds.style.zoom;
10610    WD_Zoom_1_XY.ovalue = ds.style.zoom;
10611    scaleWD(ds,oscale,nscale);
10612}
10613function WD_doZoomIN(){
10614    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10615    oscale = WD_Zoom_1_XY.value;
10616    if( oscale == 0 || oscale == '' ){
10617        oscale = 1;
10618    }
10619    nscale = oscale * zoomMag();
10620    if( 4 < nscale ){
10621        alert('maybe too large, zoom='+nscale);
10622        return;
10623    }
10624    ds.style.zoom = nscale;
10625    WD_Zoom_1_XY.value = ds.style.zoom;
10626    WD_Zoom_1_XY.ovalue = ds.style.zoom;
10627    scaleWD(ds,oscale,nscale);
10628}
10629WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
10630WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
10631
10632function WD_doResize(e){
10633    dt = WirtualDesktop_1;
10634    dt.style.width = WD_Width_1.value;
10635    dt.style.height = WD_Height_1.value;
10636    WD_Width_1.value = dt.style.width;
10637    WD_Height_1.value = dt.style.height;
10638}
10639WD_Resize_1.addEventListener('click',e => { WD_doResize(e); });
10640
10641
10642function WD_dorSResize(e){
10643    //alert('Resize Space');
10644    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10645    ds.style.width = WS_1_Width.value;
10646    ds.style.height = WS_1_Height.value;
10647    WS_1_Width.value = ds.style.width;
10648    WS_1_Height.value = ds.style.height;
10649}
10650ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
10651ds.style.width = '5120px';
10652ds.style.height = '2880px';
10653WS_1_Width.value = ds.style.width;
10654WS_1_Height.value = ds.style.height;
10655WS_1_Width.addEventListener('keydown',ignoreEvent);
10656WS_1_Width.addEventListener('keyup',ignoreEvent);
10657WS_1_Height.addEventListener('keydown',ignoreEvent);
10658WS_1_Height.addEventListener('keyup',ignoreEvent);
10659WS_Resize_1.addEventListener('click',e => { WD_dorSResize(e); });
10660
10661function WD_doScrollXY(e,sleft,stop){
10662    dt = WirtualDesktop_1;
10663    dt.scrollLeft = sleft;
10664    dt.scrollTop = stop;

```

```

10665   WD_Left_1.value = dt.scrollLeft;
10666   WD_Top_1.value = dt.scrollTop;
10667   console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
10668 }
10669 function WD_doScroll(e) {
10670   //dt = VirtualDesktop_1_Content;
10671   dt = VirtualDesktop_1;
10672   sleft = parseInt(WD_Left_1.value);
10673   stop = parseInt(WD_Top_1.value);
10674   dt.scrollLeft = sleft;
10675   dt.scrollTop = stop;
10676   WD_Left_1.value = dt.scrollLeft;
10677   WD_Top_1.value = dt.scrollTop;
10678   console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
10679 }
10680 WD_Scroll_1.addEventListener('click', e => { WD_doScroll(e); });
10681 WD_Left_1.addEventListener('keydown', ignoreEvent);
10682 WD_Left_1.addEventListener('keyup', ignoreEvent);
10683 WD_Top_1.addEventListener('keydown', ignoreEvent);
10684 WD_Top_1.addEventListener('keyup', ignoreEvent);
10685
10686 function showScrollPosition() {
10687   if( false )
10688     console.log(
10689       'wstop=' + VirtualDesktop_1.style.top + ', ' +
10690       'wsx=' + VirtualDesktop_1.style.y + ', ' +
10691       'wss=' + VirtualDesktop_1.scrollTop + ', ' +
10692       'wdtop=' + VirtualDesktop_1_Content.style.top + ', ' +
10693       'wdx=' + VirtualDesktop_1_Content.style.y + ', ' +
10694       'wds=' + VirtualDesktop_1_Content.scrollTop + ', ' +
10695     );
10696   WD_Left_1.value = VirtualDesktop_1.scrollLeft;
10697   WD_Top_1.value = VirtualDesktop_1.scrollTop;
10698 }
10699 VirtualDesktop_1.addEventListener('scroll', showScrollPosition);
10700 VirtualDesktop_1_Content.addEventListener('scroll', showScrollPosition);
10701
10702
10703 if( false ) {
10704   w = 1000 + 'px';
10705   dt.style.width = w;
10706   dt.style.height = "300px";
10707   dt.style.resize = 'both';
10708   dt.style.borderWidth = 50 + 'px';
10709   dt.style.borderRadius = 25 + 'px';
10710   console.log('--2----- #' + dt.id + ' style=' + dt.style);
10711   console.log('------- #' + dt.id + ' width=' + dt.style.width);
10712   console.log('------- #' + dt.id + ' left=' + dt.style.left);
10713   console.log('------- #' + dt.id + ' border=' + dt.style.border);
10714 }
10715
10716 function onDTResize(e) {
10717   dt = e.target;
10718   h = parseInt(dt.style.height);
10719   dt.style.borderWidth = (h * 0.075) + 'px';
10720   console.log('------- borderWidgh=' + dt.style.borderWidth);
10721 }
10722 VirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
10723 //console.log('------- #' + dt.id + ' borderWidth=' + dt.style.getProperty('border-width'));
10724
10725 settleWin(
10726   VirtualBrowser_1,
10727   VirtualBrowser_1_Location,
10728   VirtualBrowser_1_Command,
10729   VirtualBrowser_1_Frame,
10730   document.URL,
10731   500, 280, 50, 20, '#262', 1.0);
10732 settleWin(
10733   VirtualBrowser_2,
10734   VirtualBrowser_2_Location,
10735   VirtualBrowser_2_Command,
10736   VirtualBrowser_2_Frame,
10737   'https://its-more.jp/ja_jp/',
10738   500, 280, 150, 100, '#444', 1.0);
10739
10740 settleWin(
10741   VirtualBrowser_3,
10742   VirtualBrowser_3_Location,
10743   VirtualBrowser_3_Command,
10744   VirtualBrowser_3_Frame,
10745   '../gshell/gsh.go.html',
10746   'http://gshell.org/gshell/gsh.go.html',
10747   'https://golang.org',
10748   500, 280, 250, 180, '#444', 1.0);
10749   //1000, 720, 0, 0, '#444', 0.125);
10750   //1200, 720, -100, -50, '#444', 0.4);
10751
10752 function WD_ClockUpdate(e) {
10753   VirtualDesktop_1_Clock.innerHTML = DateShort();
10754   VirtualDesktop_1_Calender.innerHTML = DateHourMin();
10755 }
10756 window.setInterval(WD_ClockUpdate, 500);
10757 //GJ_Join();
10758
10759 Destroy_VirtualDesktop = function() {
10760   VirtualDesktop_1.style = "";
10761
10762   VirtualBrowser_1.removeAttribute('style');
10763   VirtualBrowser_1_Location.innerHTML = '';
10764   VirtualBrowser_1_Frame.removeAttribute('src');
10765   VirtualBrowser_1_Frame.removeAttribute('style');
10766   VirtualBrowser_1_Frame.style = "";
10767
10768   VirtualBrowser_2.removeAttribute('style');
10769   VirtualBrowser_2_Location.innerHTML = '';
10770   VirtualBrowser_2_Frame.removeAttribute('src');
10771   VirtualBrowser_2_Frame.style = "";
10772
10773   VirtualBrowser_3.removeAttribute('style');
10774   VirtualBrowser_3_Location.innerHTML = '';
10775   VirtualBrowser_3_Frame.removeAttribute('src');
10776   VirtualBrowser_3_Frame.style = "";
10777
10778   GJFactory_1.style = "";
10779   iframe0.style = "";
10780   VirtualDesktop_1.style = "";
10781 }
10782
10783 </script>
10784 <!-- VirtualDesktop } -->
10785 </details>
10786 *//</span>
10787
10788 <!-- ----- Work { ----- -->

```

```

10789//<span id="SBSidebar_WorkCodeSpan">
10790/*
10791<details><summary>SBSidebar</summary>
10792<!-- SBSidebar // 2020-0928 SatoxITS { -->
10793<h2>SBSidebar</h2>
10794<input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
10795<input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
10796<input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
10797<span id="SBSidebar_WorkCodeView"></span>
10798<script id="SBSidebar_WorkScript">
10799function SBSidebar_openWorkCodeView(){
10800    function SBSidebar_showWorkCode(){
10801        showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
10802    }
10803    SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
10804}
10805SBSidebar_openWorkCodeView(); // should be invoked by an event
10806
10807console.log('-- SbsSlider // 2020-1006-01 SatoxITS --');
10808function SetSidebar(){
10809    sidebar = document.getElementById('secondary');
10810//    console.log('primary='+primary+ 'secondary='+sidebar+ 'main='+main+ ' ');
10811    wrap = sidebar.parentNode;
10812    console.log('-- SbsSlider parent is '+wrap+', #'+wrap.id+'.'+wrap.class);
10813//wrap = wrap.parentNode;
10814//console.log('-- SbsSlider parent is '+wwrap+', #'+wwrap.id+'.'+wwrap.class);
10815//wwrap = wwrap.parentNode;
10816//console.log('-- SbsSlider parent is '+wwwrap+', #'+wwwrap.id+'.'+wwwrap.class);
10817//nsb = sidebar.cloneNode();
10818    nsb = sidebar;
10819    nsb.style.width = '100%';
10820    slider = document.createElement('div');
10821    slider.id = 'SbSlider';
10822    slider.appendChild(nsb);
10823    slider.setAttribute('class','SbSlider');
10824    nsb.style.position = 'relative';
10825    slider.style.position = 'fixed';
10826    slider.style.display = 'block'; //inline;
10827    slider.style.zIndex = 10000;
10828    // nsb.style.zIndex = 20000;
10829    nsb.style.position = 'absolute';
10830    nsb.style.minWidth = '80px';
10831    nsb.style.left = '0px';
10832    nsb.style.top = '0px';
10833
10834    w = window.innerWidth;
10835    console.log('SliderWidth '+w+: ' ,(w/3)+'px');
10836    if( w < 640 ){
10837        slider.style.setProperty('width',(w/3) + 'px','important');
10838    }
10839    main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
10840
10841    slider.style.resize = "both";
10842    slider.draggable = "true";
10843    wrap.appendChild(slider);
10844    console.log('-- added SbsSlider');
10845    //nsb.addEventListener('scroll',SbScrolled);
10846
10847    buttons = document.createElement('div');
10848    buttons.id = 'NaviButtons';
10849    buttons.setAttribute('class','NaviButtons');
10850    buttons.align = "center";
10851    buttons.innerHTML += '<'+p><a href="#TopOfPost" draggable="true">TOP<'+/a><'+/p>';
10852    buttons.innerHTML += '<'+p><a href="#EndOfPost" draggable="true">END<'+/p>';
10853    page.appendChild(buttons);
10854    buttons.style.position = 'fixed';
10855    buttons.style.zIndex = 30000;
10856    buttons.style.width = '180%';
10857    buttons.style.top = '320px';
10858    buttons.style.left = parseInt(w) * 1.0 + 'px';
10859    console.log('-- SbsSlider installed (~-)/ SatoxITS');
10860}
10861//window.addEventListener('load',SetSidebar); // after load
10862DestroyNaviButtons = function(){
10863    nb = document.getElementById('NaviButtons');
10864    if( nb != null ){
10865        nb.parentNode.removeChild(NaviButtons);
10866    }
10867}
10868</script>
10869
10870// 2020-1006 its-more.jp-blog-60000-style.css
10871<!-- {
10872<style>
10873#NaviButtons {
10874    position:fixed;
10875    display:block;
10876    width:100%;
10877    xtop:100px;
10878    xleft:10px;
10879    z-index:30000;
10880    font-size:10pt;
10881    color:#2ff !important;
10882    text-align:center;
10883    background-color:rgba(230,230,230,0.01);
10884}
10885#NaviButtons a {
10886    color:#2a2 !important;
10887    font-size:20px;
10888    text-align:center;
10889    xtext-shadow:2px 2px #8ff;
10890    resize:both;
10891    padding:6px;
10892    margin:10px;
10893    border:1px solid #288 !important;
10894    border-radius:3px;
10895    background-color:rgba(160,160,160,0.05);
10896}
10897#SbSlider {
10898    overflow:auto;
10899    resize:both !important;
10900    xoverflow-y:hidden !important;
10901    height:100px !important;
10902    display:inline !important;
10903    position:fixed !important;
10904    left:0px;
10905    top:0px;
10906    xxwidth:180px;
10907    width:24%;
10908    min-width:80px;
10909    height:100% !important;
10910    background-color:rgba(100,100,200,0.1);
10911}
10912#secondary {

```



```

10913     position:fixed;
10914     left:0px;
10915     top:0px;
10916     xxxz-index:60000;
10917     scroll-behavior: overflow !important;
10918     xxxxxxxxxxxxxxxxxxxxxxxxxxxwidth:188 !important;
10919     padding-left:4pt;
10920     color:#fff;
10921     font-size:0.5em;
10922     background-color:rgba(64,160,64,0.6) !important;
10923     white-space:nowrap;
10924 }
10925 #secondary a {
10926     color:#fff !important;
10927     text-decoration:disable !important;
10928 }
10929 #primary {
10930     position:relative;
10931     width:75% !important;
10932     left:25% !important;
10933 }
10934 #main {
10935     position:relative;
10936     width:75% !important;
10937     left:25% !important;
10938 }
10939 #site-navigation {
10940     position:relative;
10941     left:120px;
10942 }
10943 #adswsc_countertext {
10944     color:#4169e1;
10945     font-size:16pt !important;
10946     xxxfont-size:10% !important;
10947     font-weight:bold;
10948 }
10949 #nowTime {
10950     color:#a0ffa0;
10951     font-size:16pt !important;
10952     xxxfont-size:10% !important;
10953     font-weight:bold;
10954     text-shadow:1px 1px #fff;
10955 }
10956 .navigation-top {
10957     color:#22a !important;
10958     border:0px;
10959     background-color:rgba(220,220,220,0.1);
10960 }
10961
10962 .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
10963     display: block;
10964     xxwidth: 1em;
10965     xxoverflow: auto;
10966     xxheight: 1em;
10967 }
10968 .invisible-scrollbar::-webkit-scrollbar {
10969     xxdisplay: none;
10970     width:1px !important;
10971     height:1px !important;
10972 }
10973 .mostly-customized-scrollbar::-webkit-scrollbar {
10974     width: 2px;
10975     height: 2px;
10976     xxbackground-color: #aaa; xxx:or add it to the track;
10977 }
10978 .mostly-customized-scrollbar::-webkit-scrollbar-thumb {
10979     background: #000;
10980 }
10981 </style>
10982 } -->
10983
10984
10985 </details>
10986 <!-- SBSidebar_WorkCodeSpan } -->
10987 * //</span>
10988 //<!-- ----- Work } ----- -->
10989
10990
10991
10992
10993 //<!-- ----- Work { ----- -->
10994 //<span id="Template_WorkCodeSpan">
10995 /*
10996 <details><summary>Work Template</summary>
10997 <!-- ----- Template of Work// 2020-0928 SatoxITS { -->
10998 <h2>Template of Work</h2>
10999 <input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11000 <input id="Template_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11001 <input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11002 <span id="Template_WorkCodeView"></span>
11003 <script id="Template_WorkScript">
11004 function Template_openWorkCodeView(){
11005     function Template_showWorkCode(){
11006         showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
11007     }
11008     Template_WorkCodeViewOpen.addEventListener('click',Template_showWorkCode);
11009 }
11010 Template_openWorkCodeView(); // should be invoked by an event
11011 </script>
11012 </details>
11013 <!-- Template_WorkCodeSpan } -->
11014 * //</span>
11015 //<!-- ----- Work } ----- -->
11016
11017 //<script>Indexer_afterLoaded();</script>
11018 //<script>Gshell_initTopbar();</script>
11019
11020 </div>
11021 //<br></span></html>
11022

```