

```

1 /*
2 <span id="gsh" data-title="GShell" data-author="sato@its-more.jp">
3 <meta charset="UTF-8">
4 <meta name="viewport" content="width=device-width, initial-scale=1.0">
5 <link rel="icon" id="GshFaviconURL" href=""><!-- place holder -->
6 <span id="GshVersion" hidden>gsh-0.7.6-2020-10-25--SatoxITS</span>
7 <title id="GshTitle">GShell-0.7.6 by SatoxITS</title>
8
9 <div id="GshHeading">
10 <div id="GshTopbar" class="MetaWindow"></div>
11 <div id="GshPerfMon"></div>
12 <div id="GshSidebar" class="SbSidebar" style=""><div id="GshIndexer" style=""></div></div>
13 </div>
14 <div id="GshMain">
15 <div id="GshBanner" height="100px" onclick="shiftBG();">
16 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.7.6 // 2020-10-25 // SatoxITS</note></div>
17 </div>
18 </div>
19 //<!-- ----- Work { ----- -->
20 //<span id="Topbar_WorkCodeSpan">
21 /* 
22 <details><summary>Topbar</summary>
23 <!-- Topbar // 2020-1008 SatoxITS { -->
24 <h2>Topbar</h2>
25 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
26 <input id="Topbar_OpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
27 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
28 <span id="Topbar_WorkCodeView"></span>
29 </details>
30
31 <style>
32 #GshHeading {
33   display:inline;
34   overflow:visible;
35 }
36 .ConfigIcon {
37   position:absolute;
38   top:-6px;
39   left:92px;
40   width:32px;
41   height:32px;
42 }
33 .MetaWindow {
34   z-index:100;
35   position:relative;
36   display:block;
37   overflow:visible !important;
38   width:99.9%;
39   height:122px;
40   top:-22px;
41   border:1px solid #22a;
42   margin:0px;
43   left:0px;
44   line-height:1.0;
45   font-family:Georgia;
46   color:#fff;
47   font-size:12pt;
48   text-align:center;
49   vertical-align:middle;
50   padding:4px;
51   background-color:rgba(0,8,170,0.8);
52   background-color:#3d861;xxx-PBlue;
53   vertical-align:middle;
54 }
55 .MetaWindow:hover {
56   color:#000;
57   border:1px solid #22a;
58   background-color:rgba(255,255,255,1.0);
59 }
60 <script>
61 GshBanner {
62   overflow:visible;
63   display:block;
64   width:100px;
65   height:100px;
66   left:inherent !important;
67 }
68 </style>
69 <script>
70 function Topbar_openWorkCodeView(){
71   function Topbar_showWorkCode(){
72     showHTMLCode(Topbar_WorkCodeView,Topbar_WorkCodeSpan);
73   }
74   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
75 }
76 Topbar_openWorkCodeView();
77 function ConfigClick(){
78   if( 0 <= AffView.style.zIndex ) {
79     AffView.style.saved_zIndex = AffView.style.zIndex;
80     AffView.style.zIndex = -1000;
81     GshSidebar.style.zIndex = -1;
82     GshPerfMon.style.zIndex = -1;
83   }else{
84     /AffView.style.zIndex = AffView.style.saved_zIndex;
85     AffView.style.zIndex = 1;
86     GshSidebar.style.zIndex = 1;
87     GshPerfMon.style.zIndex = 1;
88     GMenu.style.zIndex = 10000000;
89   }
90   console.log('AffZIndex'+AffView.style.zIndex);
91 }
92 function Gshell_initTopbar(){
93   GshTopbar.innerHTML = GshTitle.innerHTML;
94   /<div id="ConfigIcon" class="ConfigIcon">
95   if( true ){
96     cfgi = document.createElement('img');
97     cfgi.setAttribute('class','ConfigIcon');
98     GshTopbar.appendChild(cfgi);
99     cfgi.src = ConfigICON_DATA;
100   }
101   //cfgi.style.zIndex = 1000000000;
102   //cfgi.addEventListener('click',ConfigClick);
103   GshTopbar.addEventListener('click',ConfigClick);
104 }
105 </script>
106 <!-- Topbar_WorkCodeSpan } -->
107 /* 
108 <details><summary>Indexer</summary>
109 <!-- Indexer // 2020-1007 SatoxITS { -->
110 <h2>Indexer</h2>
111 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
112 <input id="Indexer_OpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
113 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
114 <span id="Indexer_WorkCodeView"></span>
115 </details>
116 <style>
117 #gsh {
118   display:block;
119   position:scroll !important;
120   width:80% !important;
121   left:19.5% !important;
122 }
123 #GshMain {
124   z-index:1;
125   position:relative;
126   display:block;
127   width:80% !important;
128   left:19.5% !important;
129 }
130 #GshSidebar {
131   z-index:0;
132   position:relative !important;
133   position:fixed;
134   resize:both !important;
135   width:19.5% !important;
136   xoverflow-y:hidden !important;
137   xxxheight:100px !important;
138   xxxdisplay:inline !important;
139   left:0px;
140   top:0px;
141   width:19.5% !important;
142   min-width:80px;
143   xxxheight:100% !important;
144   height:0px;
145   color:#f00;
146   xxbackground-color:rgba(64,64,64,0.5);
147 }
148 
```

```

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

```

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

```

```

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

<-- Indexer_WorkCodeSpan -->

<-- //

//<!-- ----- Work ----- -->

420

421

422

423 /*

424 <h2>GShell // a General purpose Shell built on the top of Golang</h2>

425 <p>

426 <note>

427 It is a shell for myself, by myself, of myself. --SatoITS(~`)

428 prev.

429 </note>

430 </p>

431 <div id="GJFactory_x"></div>

432 </div>

433

434 Edit

435 Save

436 Load

437 Vers

438 0

439 xit

440 for

441 stop

442 infold

443 Digest

444 style="">source

445 <!-- / Pure -->

446 </div>

447 </div>

448 </div>

449 /*

450 */

451 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>

452 <h3>Fun to create a shell</h3>

453 <p>For a programmer, it must be far easy and fun to create his own simple shell

454 rightly fitting to his favor and necessities, than learning existing shells with

455 complex full features that he never use.

456 I, as one of programmers, am writing this tiny shell for my own real needs,

457 especially from scratch, with fun.

458 /*

459 For a programmer, it is fun to learn new computer languages. For long years before

460 writing this software, I had been specialized to C and early HTML2 :-).

461 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS

462 on demand as a novice of these, with fun.

463 The single file "gsh.go", that is executable by Go, contains all of the code written

464 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone

465 HTML file that works as the viewer of the code of itself, and as the "home page" of

466 this software.

467 Besides this HTML file is a Go program, you may run it as a real shell program

470 on your computer.

472 But you must be aware that this program is written under situation like above.

473 Needless to say, there is no warranty for this program in any means.

474 </p>

475 <address>Aug 2020, SatoITS (sato@its-more.jp)</address>

476 </details>

477 /*

478 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>

479 <h3>Cross-browser communication</h3>

480 <p>

483 to be written ...

484 </p>

485 <h3>Vi compatible command line editor</h3>

```

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

```

```

648 }
649 */
650
651 const {
652     NAME = "gsh"
653     VERSION = "0.7.6"
654     DATE = "2020-10-25"
655     AUTHOR = "SatoxIT5(`-`)//"
656 }
657 var
658     GSH_HOME = ".gsh" // under home directory
659     GSH_PORT = 9999
660     MaxReadSize = int64(128*1024*1024*1024) // 128GiB is too large?
661     PROMPT = "> "
662     LINESIZE = "(8*1024"
663     PATHSEP = ";" // should be ";" in Windows
664     DIRSEP = "/" // canbe '\v in Windows
665     OnWindows = false;
666 }
667 func initGshEnv(){
668     if( runtime.GOOS == "windows" ){
669         PATHSEP = ";";
670         DIRSEP = "\\" ;
671         OnWindows = true;
672     }else{
673     }
674 }
675
676 // --X logging control
677 // --A- allinfo
678 // --D- info
679 // --B- debug
680 // --T- time and resource usage
681 // --L- logfilename
682 // --E- error
683 // --F- fatal error
684 // --Nn- network
685
686 // <a name="struct">Structures</a>
687
688 // 2020-1022 Unix/Windows
689 // -----
690 //<type aStat_t syscall.Stat_t;
691 //<type aStat_t struct { syscall.Stat_t
692 type aStat_t struct {
693     Size int64
694     Mode os.FileMode
695     Rdev int64
696     Blocks int64
697     Nlink int64
698 }
699 func aStat(path string, astat *aStat_t)(error){
700     /*
701     stat := syscall.Stat_t{};
702     err := syscall.Lstat(path,&stat);
703     *astat = aStat_t(stat);
704     */
705     fi,err := os.Stat(path);
706     if( err == nil ){
707         astat.Mode = fi.Mode();
708         astat.Size = fi.Size();
709     }
710     return err;
711 }
712
713 func afStat(fd int, astat *aStat_t)(error){
714     /*
715     stat := syscall.Stat_t{};

716     err := syscall.Fstat(fd,&stat);
717     *astat = aStat_t(stat);
718     */
719     err := errors.New("NotImplemented-Fstat");
720     //fmt.Printf("----E-- fstat(%v)(%v)\n",fd,err);
721     return err;
722 }
723
724 func aAccess(path string, mode uint32)(error){
725     /*err := syscall.Access(path,mode);
726     //err := errors.New("NotImplemented-Access");
727     fi,err := os.Stat(path)
728     //fmt.Printf("----E-- Access(%v,%v)\n(%v)\n",path,mode,err);
729     if( err != nil ){
730         fmode := fi.Mode();
731         if( fmode.IsRegular() ){
732             perm := fmode.Perm();
733             if( (uint32(perm) & mode) != 0 ){
734                 return nil;
735             }
736             return errors.New("NotAccessible");
737         }
738         return errors.New("NotRegularFile");
739     }
740     */
741     return err;
742 }
743
744 // 2020-1022 Unix/Windows
745 // -----
746 type aRusage struct {
747     Syscall.Rusage
748     Utime time.Duration
749     Stime time.Duration
750     //Sys interface{}
751 }
752
753 const aRUSAGE_SELF = syscall.RUSAGE_SELF
754 const aRUSAGE_CHILDREN = syscall.RUSAGE_CHILDREN
755 /*
756 const aRUSAGE_SELF = 0
757 const aRUSAGE_CHILDREN = 1
758 func aGetrusage(sel int, ru *aRusage){
759     /*
760     sysru := syscall.Rusage();
761     syscall.Getrusage(sel,&sysru);
762     ru.Utime = time.Duration(int64(sysru.Utime.Sec)*1000000000+int64(sysru.Utime.Usec)*1000);
763     ru.Stime = time.Duration(int64(sysru.Stime.Sec)*1000000000+int64(sysru.Stime.Usec)*1000);
764     */
765     func aGetrusage(ru *aRusage, ps *os.ProcessState){
766         ru.Utime = ps.UserTime();
767         ru.Stime = ps.SystemTime();
768     }
769     func showUsage(what string,argv []string, ru *aRusage){
770         fmt.Printf(":s",what);
771         //fmt.Printf("User=%d@%06d",ru.Utime.Sec,ru.Utime.Usec)
772         //fmt.Printf("System=%d@%06d",ru.Stime.Sec,ru.Stime.Usec)
773         fmt.Printf("User=%d@%06d",ru.Utime/1000000000,(ru.Utime/1000000000)*(ru.Utime/1000000));
774         fmt.Printf("System=%d@%06d",ru.Stime/1000000000,(ru.Stime/1000000)*(ru.Stime/1000000));
775     */
776     fmt.Printf("File=%v",ru.Nwrxrs)
777     if isnin("-l",argv){
778         fmt.Printf(" Mafjl=%v",ru.Mafjfl)
779         fmt.Printf(" MaJfl=%v",ru.MaJfl)
780         fmt.Printf(" IXRSS=%v",ru.IXrss)
781         fmt.Printf(" IDRSS=%v",ru.Idrss)
782         fmt.Printf(" Nswap=%v",ru.Nswap)
783         fmt.Printf(" Read=%v",ru.Unblock)
784         fmt.Printf(" Write=%v",ru.Unblock)
785     }
786     fmt.Printf(" Snd=%v",ru.Msgsnd)
787     fmt.Printf(" Rcv=%v",ru.Msgrcv)
788     //if isnin("-l",argv) {
789     fmt.Printf(" Sig=%v",ru.Signals)
790     //}
791     */
792     fmt.Println("n");
793 }
794
795 type tCommandHistory struct {
796     Startt time.Time // command line execution started at
797     Endtt time.Time // command line execution ended at
798     Rescode int // exit code of (external) command
799     CmdErrror error // error string
800     Outdata string // output of the command
801     Fullfile []string // command - result of find
802     Rusageg [2]aRusage // Resource consumption, CPU time or so
803     Cmdid int // maybe with identified with arguments or impact
804     Workdir string // redirecton commands should not be the Cmdid
805     WorkDirx int // working directory at start
806     WorkDirx int // index in ChdirHistory
807     Cmdline string // command line
808 }
809
810 type tChdirHistory struct {
811 }

```

```

810     Dir      string
811     MovedAt   time.Time
812     CmdIndex  int
813   }
814   type CmdMode struct {
815     BackGround bool
816   }
817   type Event struct {
818     when    time.Time
819     event   int
820     evarg  int64
821     CmdIndex int
822   }
823 var CmdIndex int
824 Events []Event
825 type EPluginInfo struct {
826   Spec   *plugin.Plugin
827   Addr   plugin.Symbol
828   Name   string // maybe relative
829   Path   string // this is in Plugin but hidden
830 }
831 type GServer struct {
832   host   string
833   port   string
834 }
835
836 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
837 const ( // SumType
838   SUM_ITEMS  = 0x0000001 // items count
839   SUM_SIZE   = 0x0000002 // data length (simply added)
840   SUM_SIZEHASH = 0x0000004 // data length (hashed sequence)
841   SUM_SIZEHASH2 = 0x0000008 // sum of data (hashed sequences)
842   // also envelope attributes like time stamp can be a part of digest
843   // hashed value of sizes or mod-date of files will be useful to detect changes
844
845   SUM_WORDS  = 0x0000010 // word count is a kind of digest
846   SUM_LINES   = 0x0000020 // line count is a kind of digest
847   SUM_SUM64  = 0x0000040 // simple add of bytes, useful for human too
848
849   SUM_SUM32_BITS = 0x000100 // the number of true bits
850   SUM_SUM32_2BYTE = 0x000200 // 16bits words
851   SUM_SUM32_4BYTE = 0x000400 // 32bits words
852   SUM_SUM32_8BYTE = 0x000800 // 64bits words
853
854   SUM_SUM16_BSD = 0x010000 // UNIXsum -sum -bsd
855   SUM_SUM16_SYSV = 0x020000 // UNIXsum -sum -sysv
856   SUM_UNIXFILE  = 0x040000
857   SUM_CRCIEEE  = 0x080000
858 )
859 type CheckSum struct {
860   Files   int64 // the number of files (or data)
861   Size    int64 // content size
862   Words   int64 // word count
863   Lines   int64 // line count
864   SumType int
865   Sum64  uint64
866   Crc32Table crc32.Table
867   Crc32Val  uint32
868   Sum16  uint
869   Ctime   time.Time
870   Atime   time.Time
871   Mtime   time.Time
872   Start   time.Time
873   Done    time.Time
874   RusgAtStart [2]uintptr
875   RusgAtEnd  [2]uintptr
876 }
877 type ValueStack [][]string
878 type GshContext struct {
879   Startdir  string // the current directory at the start
880   GetLine   string // gsh-current command as a input line editor
881   CmdHistory []CommandHistory // the 1st entry is wd at the start
882   //gshPA   syscall.ProcAttr
883   gshPA   os.ProcAttr
884   CmdHistory []CommandHistory
885   CmdCurrent GCommandHistory
886   Background bool
887   BackgroundJobs []ios.ProcessState; //[]int
888   LastUsage  ausage
889   GahHomeDir string
890   TerminalId int
891   CmdTrace  bool // should be [map]
892   CmdTime   bool // should be [map]
893   PrintLines []PrintInfo
894   iValues  []string
895   iDelimeter string // field separator of print out
896   iFormat   string // default print format (of integer)
897   iValStack ValueStack
898   LastServer GServer
899   PWD    string // [gsh://host:port]
900   PWD   string // remote (target, there) working directory
901   lastCheckSum  CheckSum
902 }
903
904 func nsleep(ns time.Duration){
905   time.Sleep(ns)
906 }
907 func usleep(ns time.Duration){
908   msleep(ns*1000)
909 }
910 func msleep(ns time.Duration){
911   msleep(ns*1000000)
912 }
913 func sleep(ns time.Duration){
914   msleep(ns*1000000000)
915 }
916
917 func strBegins(str, pat string)(bool){
918   if len(pat) <= len(str){
919     yes := str[0:len(pat)] == pat
920     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,yes)
921     return yes
922   }
923   //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
924   return false
925 }
926 func isin(what string, list []string) bool {
927   for v := range list {
928     if v == what {
929       return true
930     }
931   }
932   return false
933 }
934 func isinX(what string,list[]string)(int{
935   for i,v := range list {
936     if v == what {
937       return i
938     }
939   }
940   return -1
941 }
942 func env(opts []string) {
943   env := os.Environ()
944   if isn("-s", opts){
945     sort.Slice(env,func(i,j int) bool {
946       return env[i] < env[j]
947     })
948   }
949   for _, v := range env {
950     fmt.Printf("%v\n",v)
951   }
953 }
954
955 // - rewriting should be context dependent
956 // - should postpone until the real point of evaluation
957 // - should rewrite only known notation of symbol
958 func scanInt(str string)(val int,int int){
959   leng = -1
960   for i,ch := range str {
961     if '0' <= ch && ch <= '9' {
962       leng = i+1
963     }else{
964       break
965     }
966   }
967   if 0 < leng {
968     val,_ := strconv.Atoi(str[0:leng])
969     return val,leng
970   }else{
971     return 0,0
972   }

```

```

972     }
973 }
974 func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
975     if len(str[i+1]) == 0 {
976         return 0,rstr
977     }
978     hi := 0
979     histlen := len(gshCtx.CommandHistory)
980     if str[i+1] == `!` {
981         hi = histlen - 1
982         leng = 1
983     }else{
984         hi,leng = scanInt(str[i+1:])
985         if leng == 0 {
986             return 0,rstr
987         }
988         if hi < 0 {
989             hi = histlen + hi
990         }
991     }
992     if 0 <= hi && hi < histlen {
993         var ext byte
994         if i < len(str[i+leng]) {
995             ext = str[i+leng][1]
996         }
997         //fmt.Println("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
998         if ext != `!` {
999             leng += 1
1000             xlist := [i]string{}
1001             list := gshCtx.CommandHistory[hi].FoundFile
1002             for _,v := range list {
1003                 //list[i] = escapeWhiteSP(v)
1004                 xlist = append(xlist,escapeWhiteSP(v))
1005             }
1006             //rstr += strings.Join(list, " ")
1007             rstr += strings.Join(xlist, " ")
1008         }else{
1009             if ext == `@` || ext == `d` {
1010                 // !N@ .. workdir at the start of the command
1011                 leng += 1
1012                 rstr += gshCtx.CommandHistory[hi].Workdir
1013             }else{
1014                 rstr += gshCtx.CommandHistory[hi].CmdLine
1015             }
1016         }
1017     }
1018     }
1019     return leng,rstr
1020 }
1021 func escapeWhiteSP(str string)(string){
1022     if len(str) == 0 {
1023         return "\\\z" // empty, to be ignored
1024     }
1025     rstr := ""
1026     for _,ch := range str {
1027         switch ch {
1028             case `\\`:
1029                 rstr += `\\\\`
1030             case `:`:
1031                 rstr += `\\:`
1032             case `r`:
1033                 rstr += `\\r`
1034             case `n`:
1035                 rstr += `\\n`
1036             default:
1037                 rstr += string(ch)
1038         }
1039     }
1040     return rstr
1041 }
1042 func unescapeWhiteSP(str string)(string){ // strip original escapes
1043     str := ""
1044     for i := 0; i < len(str); i++ {
1045         ch := str[i]
1046         if ch == `\\` {
1047             if i+1 < len(str) {
1048                 switch str[i+1] {
1049                     case `z`:
1050                         continue;
1051                     default:
1052                         rstr += string(ch)
1053                 }
1054             }
1055         }
1056     }
1057     return rstr
1058 }
1059 func unescapeWhiteSPv(strv []string)([]string){ // strip original escapes
1060     strv := []string{}
1061     for _,v := range strv {
1062         strv = append(strv,unescapeWhiteSP(v))
1063     }
1064     return strv
1065 }
1066 // <a name="comexpansion">str-expansion</a>
1067 // - this should be a macro processor
1068 func substSubst(gshCtx *GshContext,str string,histonly bool) string {
1069     rbuf := []byte{}
1070     if false{
1071         // Unicode should be cared as a character
1072         return str
1073     }
1074     //rstr := ""
1075     inEsc := 0 // escape character mode
1076     for i := 0; i < len(str); i++ {
1077         ch := str[i]
1078         if ch == `@` {
1079             if i+1 < len(str) {
1080                 if i+1 < len(str) {
1081                     switch str[i+1] {
1082                         case `z`:
1083                             continue;
1084                         default:
1085                             rbuf = append(rbuf,[byte(rs)..])
1086                         i += len(rs)
1087                         //rstr = xrstr
1088                         continue;
1089                     }
1090                 }
1091             }
1092             switch inEsc {
1093                 case `\\`:
1094                     switch ch {
1095                         case `\\`:
1096                             case `s`:
1097                                 case `t`:
1098                                 case `r`:
1099                                 case `n`:
1100                                 case `z`:
1101                                     inEsc = 0; continue // empty, to be ignored
1102                                 }
1103                     inEsc = 0
1104                 case `%`:
1105                     switch {
1106                         case ch == `s`:
1107                             case ch == `t`:
1108                             //rstr = rstr + time.Now().Format(time.Timestamp)
1109                             rs := time.Now().Format(time.Timestamp)
1110                             rbuf = append(rbuf,[byte(rs)..])
1111                             inEsc = 0
1112                             continue;
1113                         default:
1114                             // postpone the interpretation
1115                             //rstr = rstr + "%" + string(ch)
1116                             rbuf = append(rbuf,ch)
1117                             inEsc = 0
1118                             continue;
1119                         }
1120                     inEsc = 0
1121                 }
1122             }
1123             rbuf = append(rbuf,ch)
1124         }
1125         //fmt.Println("--D--subst(%s)(%s)\n",str,string(rbuf))
1126         return string(rbuf)
1127     }
1128 }
1129 func showFileInfo(path string,opts []string) {
1130     if isnin("-l",opts) || isnin("-ls",opts) {
1131         fi, err := os.Stat(path)
1132         if err != nil {
1133             fmt.Printf("----- (%v)",err)

```

```

1134     }else{
1135         mod := fi.ModTime()
1136         date := mod.Format(time.Stamp)
1137         fmt.Printf("%v %s",fi.Mode(),date)
1138     }
1139 }
1140 fmt.Println("#",path)
1141 if isnin("-sp",opts) {
1142     fmt.Println(" ")
1143 }else{
1144     if ! isnin("-n",opts) {
1145         fmt.Println("\n")
1146     }
1147 }
1148 func userHomeDir()(string,bool){
1149 /*
1150 homedir,_ = os.UserHomeDir() // not implemented in older Golang
1151 */
1152 homedir,found := os.LookupEnv("HOME")
1153 //fmt.Println("--> HOME=%v(%n)",homedir,found)
1154 if !found {
1155     return "/tmp",found
1156 }
1157 return homedir,found
1158 }
1159
1160 func toFullPath(path string) (fullpath string) {
1161     if path[0] == '/' {
1162         return path
1163     }
1164     pathv := strings.Split(path,DIRSEP)
1165     switch len(pathv) {
1166     case pathv[0] == ":":
1167         pathv[0] = os.Getenv("HOME")
1168     case pathv[0] == ".": // all ones should be interpreted
1169         cwd, _ := os.Getwd()
1170         ppathv := strings.Split(cwd,DIRSEP)
1171         pathv[0] = strings.Join(ppathv,DIRSEP)
1172     case pathv[0] == userHomeDir():
1173         pathv[0] = userHomeDir()
1174     default:
1175         cwd, _ := os.Getwd()
1176         pathv[0] = cwd + DIRSEP + pathv[0]
1177     }
1178     return strings.Join(pathv,DIRSEP)
1179 }
1180
1181 func IsRegFile(path string)(bool){
1182     fi, err := os.Stat(path)
1183     if err != nil {
1184         fm := fi.Mode()
1185         return fm.IsRegular()
1186     }
1187     return false
1188 }
1189
1190 // <a name="encode">Encode / Decode</a>
1191 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
1192 func (gshctx *GshContext)Enc(argv[]string){
1193     file := os.Stdin
1194     buff := make([]byte,LINELENGTH)
1195     li := 0
1196     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
1197     for li = 0; ; li++ {
1198         count, err := file.Read(buff)
1199         if count <= 0 {
1200             break
1201         }
1202         if err != nil {
1203             break
1204         }
1205         encoder.Write(buff[0:count])
1206     }
1207     encoder.Close()
1208 }
1209 func (gshctx *GshContext)Dec(argv[]string){
1210     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
1211     li := 0
1212     buff := make([]byte,LINELENGTH)
1213     for li = 0; ; li++ {
1214         count, err := decoder.Read(buff)
1215         if count <= 0 {
1216             break
1217         }
1218         if err != nil {
1219             break
1220         }
1221         os.Stdout.Write(buff[0:count])
1222     }
1223 }
1224 // lnsp [N] [<crlf>][C \n]
1225 func (gshctx *GshContext)SplitLine(argv[]string){
1226     strRep := isnin("-str",argv) // "...+
1227     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1228     ni := 0
1229     toI := 0
1230     for ni = 0; ; ni++ {
1231         line, err := reader.ReadString('\n')
1232         if len(line) <= 0 {
1233             if err != nil {
1234                 fmt.Fprintf(os.Stderr,"--I-- lnsp %d to %d (%v)\n",ni,toI,err)
1235                 break
1236             }
1237         }
1238         off := 0
1239         ilen := len(line)
1240         remlen := len(line)
1241         if strRep { os.Stdout.WriteString([]byte("\n")) }
1242         for oi := 0; 0 < remlen; oi++ {
1243             olen := remlen
1244             addnl := false
1245             if 72 < olen {
1246                 olen /= 72
1247                 addnl = true
1248             }
1249             fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d\n",
1250                         toI,ni,oi,off,olen,remlen,ilen)
1251             toI += olen
1252             os.Stdout.WriteString([]byte(line[0:olen]))
1253             if addnl {
1254                 if strRep {
1255                     os.Stdout.WriteString([]byte("\n"))
1256                 }else{
1257                     os.Stdout.WriteByte(0x0d)
1258                     os.Stdout.WriteByte(0x0a)
1259                     os.Stdout.WriteString([]byte("\n"))
1260                 }
1261             }
1262             line = line[olen:]
1263             off += olen
1264             remlen -= olen
1265         }
1266         if strRep { os.Stdout.WriteString([]byte("\n")) }
1267     }
1268     fmt.Fprintf(os.Stderr,"--I-- lnsp %d to %d\n",ni,toI)
1269 }
1270
1271 // CRC32 <a href="https://golang.jp/pkg/hash-crc32">crc32</a>
1272 // 1 0000 0100 1100 0001 0001 1101 0111
1273 var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
1274 var CRC32NET uint32 = uint32(0x04C11DB7) // Net cksum
1275 func bytencrc32add(crc uint32,si[]byte,len uint64)(uint32){
1276     var ol uint64
1277     for ol = 0; ol < len; ol++ {
1278         var oct = str[ol]
1279         for bi := 0; bi < 8; bi++ {
1280             crc = crc ^ byte(crc<<1)^ byte((ol&1)&oct)
1281             if ((crc & 0x80000000) != 0) {
1282                 ovf1 := (oct & 0x80) != 0
1283                 ovf2 := (oct & 0x080) != 0
1284                 ovf := (ovf1 && ovf2) || (!ovf1 && ovf2)
1285                 oct <<= 1
1286                 crc <<= 1
1287             }
1288         }
1289     }
1290     //printf(stderr,"--CRC32 return %d\n",crc,len)
1291     return crc;
1292 }
1293 func bytencrc32end(crc uint32, len uint64)(uint32){
1294     var silen = make([]byte,4)
1295     var li = 0
1296     for li = 0; li < 4; {

```

```

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

```

```

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

```

```

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

```

```

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

```

```

1944     if name == "gsh.ppid" {
1945         return true, os.Getppid()
1946     }
1947     return false, 0
1948 }
1949
1950 func echo(argv []string, nlen bool){
1951     for ai := 1; ai < len(argv); ai++ {
1952         if i < ai {
1953             fmt.Printf(" ")
1954         }
1955         arg := argv[ai]
1956         found, val := getval(arg)
1957         if found {
1958             fmt.Printf("%d",val)
1959         }else{
1960             fmt.Printf("%s",arg)
1961         }
1962     }
1963     if nlen {
1964         fmt.Printf("\n");
1965     }
1966 }
1967
1968 func resfile() string {
1969     return "gsh.tmp"
1970 }
1971 //var refF *File
1972 func remap() {
1973     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
1974     // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
1975     _, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
1976     if err != nil {
1977         fmt.Printf("refF could not open: %s\n",err)
1978     }else{
1979         fmt.Printf("refF opened\n")
1980     }
1981 }
1982
1983 // @@2020-0821
1984 func gshScanArg(str string,strip int)(argv []string){
1985     var si = 0
1986     var sb = 0
1987     var inBracket = 0
1988     var argl = make([]byte,LINESIZE)
1989     var ax = 0
1990     debug := false
1991
1992     for ; si < len(str); si++ {
1993         if str[si] != ' ' {
1994             break
1995         }
1996     }
1997     sb = si
1998     for ; si < len(str); si++ {
1999         if sb < si {
2000             if debug {
2001                 fmt.Printf("--Da- %d %2d-%2d %s ... %s\n",
2002                     inBracket,sb,si,argl[0:ax],str[si:])
2003             }
2004         ch := str[si]
2005         if ch == '(' {
2006             inBracket += 1
2007             if 0 < strip && inBracket <= strip {
2008                 //fmt.Println("stripLEV #d <= #d\n",inBracket,strip)
2009                 continue
2010             }
2011         }
2012         if 0 < inBracket {
2013             if ch == ')' {
2014                 inBracket -= 1
2015                 if 0 < strip && inBracket <= strip {
2016                     //fmt.Println("stripLEV #d <= #d\n",inBracket,strip)
2017                     continue
2018                 }
2019             }
2020         argl[sx] = ch
2021         sx += 1
2022         continue
2023     }
2024     if str[si] == ' ' {
2025         argv = append(argv,str(string(argl[0:ax])))
2026         if debug {
2027             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2028                 -1+len(argv),sb,si,str[sb:si],string(str[si:]))
2029         }
2030         sb = si+1
2031         ax = 0
2032         continue
2033     }
2034     argl[ax] = ch
2035     ax += 1
2036 }
2037
2038 if sb < si {
2039     argv = append(argv,str(string(argl[0:ax])))
2040     if debug {
2041         fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2042             -1+len(argv),sb,si,string(argl[0:ax]),string(str[si:]))
2043     }
2044 }
2045 if debug {
2046     fmt.Printf("--Da- %d [%s] > [%d]%%\n",strip,str,len(argv),argv)
2047 }
2048 return argv
2049 }
2050
2051 // should get stderr (into tmpfile ?) and return
2052 func (gsh*Context)Open(name,mode string)(pin*os.File,pout*os.File,err bool){
2053     //var pv = [jint-1,-1]
2054     //syscall.Pipe(pv)
2055
2056     xargs := gshScanArg(name,1)
2057     name = strings.Join(xargs, " ")
2058
2059     //pin = os.NewFile(uintptr(pv[0]),"StdoutOf-["+name+"]")
2060     //pout = os.NewFile(uintptr(pv[1]),"StdinOf-["+name+"]")
2061     pin,pout,_ = os.Pipe();
2062
2063     fidx := 0
2064     dir := "?"
2065     if mode == "r" {
2066         dir = "<"
2067     }
2068     fidx = 1 // read from the stdout of the process
2069     if mode == "w" {
2070         dir = ">"
2071         fidx = 0 // write to the stdin of the process
2072     }
2073     gshPA := gsh.gshPA
2074     savfd := gshPA.Files[fidx]
2075
2076     var fd uintptr = 0
2077     if mode == "r" {
2078         //fd = pout.Fd()
2079         //gshPA.Files[fidx] = pout.Fd()
2080         gshPA.Files[fidx] = pout;
2081     }else{
2082         //fd = pin.Fd()
2083         //gshPA.Files[fidx] = pin.Fd()
2084         gshPA.Files[fidx] = pin;
2085     }
2086     // should do this by Goroutine?
2087     if false {
2088         fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
2089         fmt.Printf("--REDI [%d,%d,%d]>[%d,%d,%d]\n",
2090             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
2091             pin.Fd(),pout.Fd(),pout.Fd())
2092
2093         savi := os.Stdin
2094         save := os.Stdout
2095         save := os.Stderr
2096         os.Stdout = pin
2097         os.Stderr = pout
2098         gsh.BackGround = true
2099         gsh.gshellLlh(name)
2100         gsh.BackGround = false
2101         os.Stdin = savi
2102         os.Stdout = save
2103         os.Stderr = save
2104     }
2105     gshPA.Files[fidx] = savfd

```

```

2106     return pin,pout,false
2107   }
2108
2109 // <a name="ex-commands">External commands</a>
2110 func (gsh*GahContext)execCommand(exec bool, argv []string) (notf bool,exit bool) {
2111   if gsh.CmdTrace { fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }
2112
2113   gshPA := gsh.gshPA
2114   fullpath, itis := which("PATH",[]string{"which",argv[0],"-s"})
2115   if itis == false {
2116     return true,false
2117   }
2118   fullpath := fullpath[0]
2119   argv = unescapePath(argv)
2120   if 0 < len(argv)-1 {
2121     argv := argv[0:len(argv)-1]
2122     if argv[0] == "which" {
2123       if itis == false {
2124         fmt.Printf("-F-- Go not found\n")
2125         return false,true
2126       }
2127       gofullpath := gofullpath[0]
2128       argv = unescapePath(argv)
2129       fmt.Printf("-I-- %s %s\n",gofullpath,
2130                   argv[0],argv[1],argv[2])
2131       if exec {
2132         syscall.Exec(gofullpath,argv,os.Environ())
2133       }else{
2134         pid, _ := syscall.ForkExec(gofullpath,argv,&gshPA);
2135         proc,_ := os.StartProcess(gofullpath,argv,&gshPA);
2136         pstat,_ := proc.Wait();
2137         pid = pstat.Pid();
2138         if gsh.BackGround {
2139           fmt.Fprintf(stderr,"--Ip- in Background pid(%d)%d(%v)\n",pid,len(argv),argv)
2140           gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2141         }
2142       }else{
2143         rusage := aRusage {}
2144         syscall.Wait4(pid,nil,0,&rusage)
2145         gsh.LastUsage = rusage
2146         gsh.CmdCurrent.Rusagev[1] = rusage
2147       }
2148     }
2149   }
2150   gsh.LastUsage = *pstat.SysUsage().(*aRusage);
2151   gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2152   /*
2153    *setrusage(&gsh.LastRusage,pstat);
2154    *gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2155    */
2156   else{
2157     if exec {
2158       syscall.Exec(fullpath,argv,os.Environ())
2159     }else{
2160       //pid, _ = syscall.ForkExec(fullpath,argv,&gshPA);
2161       proc,_ := os.StartProcess(fullpath,argv,&gshPA);
2162       pstat,_ := proc.Wait();
2163       pid := pstat.Pid();
2164       //fmt.Printf("%d(%v)\n",pid); // '&' to be background
2165     if( false )
2166     fmt.Printf("Sys=%v\n",gshPA.Sys);
2167     if( gshPA.Sys != nil ){
2168       //fmt.Println("inFG=%v\n",gshPA.Sys.Foreground);
2169     }
2170     if gsh.BackGround {
2171       fmt.Fprintf(stderr,"--Ip- in Background pid(%d)%d(%v)\n",pid,len(argv),argv)
2172       gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2173     }
2174   }
2175   else{
2176     rusage := aRusage {}
2177   }
2178   /*
2179   gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2180   gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2181   */
2182   aSetrusage(&gsh.LastRusage,pstat);
2183   gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2184   /*
2185   */
2186   else{
2187     if exec {
2188       syscall.Exec(fullpath,argv,os.Environ())
2189     }else{
2190       //pid, _ = syscall.ForkExec(fullpath,argv,&gshPA);
2191       proc,_ := os.StartProcess(fullpath,argv,&gshPA);
2192       pstat,_ := proc.Wait();
2193       pid := pstat.Pid();
2194       //fmt.Printf("%d(%v)\n",pid); // '&' to be background
2195     if( false )
2196     fmt.Println("Sys=%v\n",gshPA.Sys);
2197     if( gshPA.Sys != nil ){
2198       //fmt.Println("inFG=%v\n",gshPA.Sys.Foreground);
2199     }
2200     if gsh.BackGround {
2201       fmt.Fprintf(stderr,"--Ip- in Background pid(%d)%d(%v)\n",pid,len(argv),argv)
2202       gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
2203     }
2204   }
2205   else{
2206     rusage := aRusage {}
2207   }
2208   /*
2209   gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2210   gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2211   */
2212   aSetrusage(&gsh.LastRusage,pstat);
2213   gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2214   /*
2215   */
2216   return false,false
2217 }
2218
2219 // <a name="builtin">Builtin Commands</a>
2220 func (gshctx *GshContext) sleep(argv []string) {
2221   if len(argv) < 2 {
2222     fmt.Println("Sleep 100ms, 100us, 100ns, ...\n")
2223     return
2224   }
2225   duration := argv[1];
2226   err := time.ParseDuration(duration)
2227   if err != nil {
2228     d, err = time.ParseDuration(duration+"s")
2229     if err != nil {
2230       fmt.Println("duration ? %s (%s)\n",duration,err)
2231       return
2232     }
2233   }
2234   /*
2235   */
2236   //fmt.Println("Sleep %v\n",duration)
2237   time.Sleep(d)
2238   if 0 < len(argv[2:]) {
2239     gshctx.gshelv(argv[2:])
2240   }
2241 }
2242 func (gshctx *GshContext) repeat(argv []string) {
2243   if len(argv) < 2 {
2244     return
2245   }
2246   start0 := time.Now()
2247   for ri,_ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
2248     if 0 < len(argv[2:]){
2249       //start := time.Now()
2250       gshctx.gshelv(argv[2:])
2251       end := time.Now()
2252       elps := end.Sub(start0);
2253       if( 1000000000 < elps ){
2254         fmt.Println("(repeat)%d %v)\n",ri,elps);
2255       }
2256     }
2257   }
2258 }
2259 func (gshctx *GshContext) gen(argv []string) {
2260   gshPA := gshctx.gshPA
2261   if len(argv) < 2 {
2262     fmt.Println("Usage: %s N\n",argv[0])
2263     return
2264   }
2265   // should br repeated by "repeat" command
2266   count := strconv.Atoi(argv[1])
2267   //fd := gshPA.Files[1] // Stdout
2268   //file := os.NewFile(fd,"internalstdout")
2269   file := gshPA.Files[1]; // Stdout
2270   file.Truncate(0).Gen. Count=fd to [%d]\n",count,file.Fd())
2271   //buf := [byte]r
2272   outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
2273   for gi := 0; gi < count; gi++ {
2274     file.WriteString(outdata)
2275   }
2276   //file.WriteString("\n")
2277   fmt.Println("\n%d B\n",count*len(outdata));
2278   //file.Close()
2279 }
2280
2281 // <a name="remote">Remote Execution</a> // 2020-0820
2282 func (gshctx *GshContext) remote(argv []string){
2283   elps := time.Now().SubFrom()
2284   if 1000000000 < elps {
2285     return fmt.Sprintf("%#d.%#02ds",elps/1000000000,(elps%1000000000)/1000000)
2286   }else{
2287     if 1000000 < elps {
2288       return fmt.Sprintf("[%d.%#03dms]",elps/1000000,(elps%1000000)/1000)
2289     }else{
2290       return fmt.Sprintf("[%d.%#03dus]",elps/1000,(elps%1000))
2291     }
2292 }

```

```

2268 }
2269 //func abstime(nanos int64)(string){
2270 func abstime(nanos time.Duration)(string){
2271     if 1000000000 < nanos {
2272         return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%100000000)/10000000)
2273     }else{
2274         if 100000 < nanos {
2275             return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
2276         }else{
2277             return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
2278         }
2279     }
2280 func abssize(size int64)(string){
2281     fsize := float64(size)
2282     if 1024*1024*1024 < size {
2283         return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
2284     }else{
2285         if 1024*1024 < size {
2286             return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
2287         }else{
2288             return fmt.Sprintf("%.3fKiB",fsize/1024)
2289         }
2290     }
2291 func abssize(size int64)(string){
2292     fsize := float64(size)
2293     if 1024*1024*1024 < size {
2294         return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
2295     }else{
2296         if 1024*1024 < size {
2297             return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
2298         }else{
2299             return fmt.Sprintf("%.3fKiB",fsize/1024)
2300         }
2301 }
2302 func abbspeed(totalB int64,ns int64)(string){
2303     MBS := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2304     if 1000 <= MBS {
2305         return fmt.Sprintf("%.3fGB/s",MBS/1000)
2306     }else{
2307         if 1 <= MBS {
2308             return fmt.Sprintf("%.3fMB/s",MBS)
2309         }else{
2310             return fmt.Sprintf("%.3fKB/s",MBS*1000)
2311         }
2312     }
2313 func abbspeed(totalB int64,ns time.Duration)(string){
2314     MBS := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2315     if 1000 <= MBS {
2316         return fmt.Sprintf("%.3fGBps",MBS/1000)
2317     }else{
2318         if 1 <= MBS {
2319             return fmt.Sprintf("%.3fMBps",MBS)
2320         }else{
2321             return fmt.Sprintf("%.3fKBps",MBS*1000)
2322         }
2323     }
2324 func fileRelay(what string,in*os.File,out*os.File,size int64,bsz int)(wcount int64){
2325     Start := time.Now()
2326     buff := make([]byte,bsz)
2327     var total int64 = 0
2328     var rem int64 = size
2329     nio := 0
2330     Prev := time.Now()
2331     var PreSize int64 = 0
2332     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) START\n",
2333             what,absize(total),size,nio)
2334     for i:= 0 ; i++ {
2335         var len = bsz
2336         if int(rem) < len {
2337             len = int(rem)
2338         }
2339         Now := time.Now()
2340         Elps := Now.Sub(Prev);
2341         if 1000000000 < Now.Sub(Prev) {
2342             fmt.Println(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %s\n",
2343             what,absize(total),size,lio,
2344             abbspeed((total-PreSize),Elps))
2345         }
2346         Prev = Now;
2347         PreSize = total
2348         rlen := len
2349         if in != nil {
2350             // should watch the disconnection of out
2351             rcc,err := in.Read(buff[0:rlen])
2352             if err != nil {
2353                 fmt.Println(Elapsed(Start)+"--En- X: %s read(%v,%v)<%v\n",
2354                     what,rcc,err,in.Name())
2355                 break
2356             }
2357             rlen = rcc
2358             if string(buff[0:10]) == "((SoftEOF " {
2359                 var ecc int
2360                 fmt.Scanf(string(buff),"(SoftEOF %v",&ecc)
2361                 fmt.Println(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))%v\n",
2362                     what,ecc,total)
2363                 if ecc == total {
2364                     break
2365                 }
2366             }
2367         }
2368         wlen := rlen
2369         if out != nil {
2370             wcc,err := out.Write(buff[0:rlen])
2371             if err != nil {
2372                 fmt.Println(Elapsed(Start)+"--En- X: %s write(%v,%v)>%v\n",
2373                     what,wcc,err,out.Name())
2374                 break
2375             }
2376             wlen = wcc
2377         }
2378         if wlen < rlen {
2379             fmt.Println(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2380                     what,wlen,rlen)
2381             break;
2382         }
2383         nio += 1
2384         total += int64(rlen)
2385         rem -= int64(rlen)
2386         if rem <= 0 {
2387             break
2388         }
2389     }
2390     Done := time.Now()
2391     Elps := float64(Done.Sub(Start))/1000000000 //Seconds
2392     Total := float64(total)/1000000 //MB
2393     Mbs := Total/Elps
2394     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) tv %.3fMB/s\n",
2395             what,total,nio,absize(total),Mbs)
2396     return total
2397 }
2398 func tcppush(clnt *os.File){
2399     // shrink socket buffer and recover
2400     usleep(100);
2401 }
2402 func (gsh*GshContext)ReexecServer(argv[]string{
2403     debug := true
2404     Start := time.Now()
2405     Start := Start0
2406     if local == ":" { local = "0.0.0.0:9999" }
2407     local := "0.0.0.0:9999"
2408     if 0 < len(argv) {
2409         if argv[0] == "-s" {
2410             debug = false
2411             argv = argv[1:]
2412         }
2413     }
2414     if 0 < len(argv) {
2415         argv = argv[1];
2416     }
2417     port, err := net.ResolveTCPAddr("tcp",local);
2418     if err != nil {
2419         fmt.Println("--En- S: Address error: %s(%s)\n",local,err)
2420         return
2421     }
2422     sconn, err := net.ListenTCP("tcp", port)
2423     if err != nil {
2424         fmt.Println(Elapsed(Start)+"--In- S: Listening at *s...\n",local);
2425     }
2426     sconn, err := net.ListenTCP("tcp", port)
2427     if err != nil {
2428

```

```

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

```

```

2592
2593 dport, err := net.ResolveTCPAddr("tcp",remote);
2594 if err != nil {
2595     fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
2596     return -1,AddressError
2597 }
2598 fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
2599 serv, err := net.DialTCP("tcp",nil,dport)
2600 if err != nil {
2601     fmt.Printf(Elapsed(Start)+"--Connection error: %s (%s)\n",remote,err)
2602     return -1,ConnectError
2603 }
2604 if debug {
2605     al := serv.LocalAddr()
2606     fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
2607 }
2608
2609 req = ""
2610 res := make([]byte,LINESIZE)
2611 count,err := serv.Read(res)
2612 if err != nil {
2613     fmt.Printf("--En- S: (%#d,%v) %v",count,err,string(res))
2614 }
2615 if debug { fmt.Println(Elapsed(Start)+"--In- S: %v",string(res)) }
2616
2617 if argv[0] == "GET" {
2618     savePA := gshGetPath
2619     var bsSize int = 32*1024
2620     var fname string = argv[1]
2621     var fsize int64 = 0
2622     var pid int = -1
2623     fmt.Scanf(string(res),"#d %d",&rcode,&dsize)
2624     fmt.Println(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
2625     if 3 < len(argv) {
2626         fname = argv[2]
2627         if strings.HasPrefix(fname,"{") {
2628             xin,xout,err := gsh.Popen(fname,"w")
2629             if err {
2630                 xin.Close()
2631                 defer xout.Close()
2632                 out = xout
2633                 out_tobeclosed = xout
2634                 pid = 0 // should be its pid
2635             }
2636         } else {
2637             // should write to temporary file
2638             // should suppress 'C on tty'
2639             xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0666)
2640             if err != nil {
2641                 fmt.Println("--En- %v\n",err)
2642             }
2643             out = xout
2644             //fmt.Println("--In-- %d > %s\n",out.Fd(),fname)
2645         }
2646     }
2647     in := serv.File()
2648     fileRelay("RecvGET",in,out,dsize,bsize)
2649     if err != nil {
2650         gshPA = savePA // recovery of FD(), and more?
2651         fmt.Println(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
2652         out_tobeclosed.Close()
2653         //syscall.Wait4(pid,nil,0,nil) //@@
2654     }
2655 }
2656
2657 else if argv[0] == "PUT" {
2658     remote, _ := serv.File()
2659     var local *os.File = nil
2660     var dsizr int64 = 32*1024*1024
2661     var dsize int = 64*1024
2662     var ofile string = "."
2663     //fmt.Println("--I- Rex %v\n",argv)
2664     if 1 < len(argv) {
2665         fname := argv[1]
2666         if strings.HasPrefix(fname,"-z") {
2667             fmt.Sscanf(fname[2],"#d",&dsize)
2668         } else {
2669             if strings.HasPrefix(fname,"{") {
2670                 xin,xout,err := gsh.Popen(fname,"r")
2671                 if err {
2672                     xout.Close()
2673                     defer xin.Close()
2674                     /in = xin
2675                     local = xin
2676                     fmt.Println("--In- [td] < Upload output of %v\n",
2677                           local.Name())
2678                     ofile = fname
2679                     dsize = MacStreamSize
2680                 }
2681             } else {
2682                 xlocal,err := os.Open(fname)
2683                 if err != nil {
2684                     fmt.Println("--En- (%s)\n",err)
2685                     local = nil
2686                 } else {
2687                     local = xlocal
2688                     fi, err := local.Stat()
2689                     dsize = fi.Size()
2690                     defer local.Close()
2691                     //fmt.Println("--I-- Rex in(%v / %v)\n",ofile,dsize)
2692                 }
2693                 ofile = fname
2694                 fmt.Println(Elapsed(Start)+"--In- L: open(%v,r)=%v (%v)\n",
2695                           fname,dsizr,local,err)
2696             }
2697         }
2698     }
2699     if 2 < len(argv) && argv[2] != "-z" {
2700         ofile = argv[2]
2701     }
2702     //fmt.Println("(fd)%v B.ofile=%v\n",len(argv),argv,ofile)
2703 }
2704 //fmt.Println(Elapsed(Start)+"--In- R: filePut(%v,%v,%v,%v)",remote,remote,dsize,bsize)
2705
2706 fmt.Println(Elapsed(Start)+"--In- PUT %v (%v)\n",dsize,bsize)
2707 if dsize > 0 { fmt.Println(Elapsed(Start)+"--In- C: %v",req) }
2708 fmt.Println(Elapsed(Start)+"--In- R: filePut(%v,%v,%v,%v)",remote,remote,dsize,bsize)
2709
2710 if debug { fmt.Println(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
2711 fileRelay("SendPUT",remote,remote,dsize,bsize)
2712
2713 if debug { fmt.Println(Elapsed(Start)+"--In- R: filePut(%v,%v,%v,%v)",remote,remote,dsize,bsize)
2714 if err != nil {
2715     fmt.Println(Elapsed(Start)+"--En- (%s)\n",err)
2716     local = nil
2717 } else {
2718     local = xlocal
2719     fi, err := local.Stat()
2720     dsize = fi.Size()
2721     defer local.Close()
2722     //fmt.Println("--I-- Rex in(%v / %v)\n",ofile,dsize)
2723     ofile = fname
2724     fmt.Println(Elapsed(Start)+"--In- L: open(%v,r)=%v (%v)\n",
2725                           fname,dsizr,local,err)
2726 }
2727
2728 if 2 < len(argv) && argv[2] != "-z" {
2729     ofile = argv[2]
2730 }
2731 //fmt.Println(Elapsed(Start)+"--In- R: filePut(%v,%v,%v,%v)",remote,remote,dsize,bsize)
2732
2733 if debug { fmt.Println(Elapsed(Start)+"--In- R: filePut(%v,%v,%v,%v)",remote,remote,dsize,bsize)
2734 if err != nil {
2735     fmt.Println(Elapsed(Start)+"--En- (%s)\n",err)
2736 } else {
2737     res = string(res[:count])
2738 }
2739 if err != nil {
2740     fmt.Println(Elapsed(Start)+"--En- S: (%d,%v) %v",rcode,err,res)
2741 }
2742 serv.Close()
2743 /conn.Close()
2744
2745 var stat string
2746 var rcode int
2747 var rcode,err,stat
2748 /fmt.Println("--D- Client: tv (%v)",rcode,stat)
2749 return rcode,res
2750
2751 // <a name="remote_sh">Remote Shell</a>
2752 func RemoteSh(args ...string) {
2753     func(gsh*Context)fileCopy(argv[]string){
2754         var host = ""

```

```

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

```

```

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

```

```

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

```

```

3240     if len(argv) <= 1 {
3241         dir = toFullPath("~")
3242     } else {
3243         i := skipopts(argv[1:])
3244         if i < 0 {
3245             dir = toFullPath("~")
3246         } else {
3247             dir = argv[i+1]
3248         }
3249     }
3250     if strBegins(dir, "-") {
3251         if dir == "-0" { // obsolete
3252             dir = gshCtx.StartDir
3253         } else {
3254             if dir == "-i" {
3255                 index := len(cdhist) - 1
3256                 if 0 < index { index -= 1 }
3257                 dir = cdhist[index].dir
3258             } else {
3259                 index, err := strconv.Atoi(dir[1:])
3260                 if err != nil {
3261                     fmt.Printf("--E-- xChdir(%v)\n", err)
3262                     dir = "?"
3263                 } else {
3264                     if len(ghctx.ChdirHistory) <= index {
3265                         fmt.Printf("--E-- xChdir(history range error)\n")
3266                         dir = "?"
3267                     } else {
3268                         dir = cdhist[index].dir
3269                     }
3270                 }
3271             }
3272             if dir == "?" {
3273                 err := os.Chdir(dir)
3274                 if err != nil {
3275                     fmt.Printf("--E-- xChdir(%s)(%v)\n", argv[1], err)
3276                 } else {
3277                     cwd, _ := os.Getwd()
3278                     if cwd != nil {
3279                         hist1 := GchdirHistory {
3280                             hist1.dir = cwd
3281                             hist1.MovedAt = time.Now()
3282                             hist1.Command = argv[1] + (ghctx.CommandHistory)+1
3283                             ghctx.ChdirHistory = append(cdhist,hist1)
3284                             if !isIn("-s", argv) {
3285                                 // cwd, _ := os.Getwd()
3286                                 // fmt.Println("%s\n", cwd)
3287                                 ix := len(ghctx.ChdirHistory)-1
3288                                 gshCtx.ShowChdirHistory(ix,hist1,argv)
3289                             }
3290                         }
3291                     }
3292                 if isIn("-ls", argv) {
3293                     cwd, _ := os.Getwd()
3294                     showFileInfo(cwd,argv);
3295                 }
3296             }
3297         }
3298     func TimeValSub(tv1 time.Duration, tv2 time.Duration) {
3299         //tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
3300         tv1 -= tv2;
3301     }
3302     func RusageSub(ru1, ru2 [2]aRusage)([2]aRusage){
3303         TimeValSub(&ru1[0].Utime,&ru2[0].Utime)
3304         TimeValSub(&ru1[0].Stime,&ru2[0].Stime)
3305         TimeValSub(&ru1[1].Utime,&ru2[1].Utime)
3306         TimeValSub(&ru1[1].Stime,&ru2[1].Stime)
3307         return ru1
3308     }
3309     func TimeValAdd(tv1 time.Duration, tv2 time.Duration) time.Duration{
3310         //tvs := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
3311         tv1 += tv2;
3312         return tvs;
3313     }
3314     /*
3315     func RusageAdd(ru1, ru2 [2]aRusage)([2]aRusage){
3316         TimeValAdd(&ru1[0].Utime,ru2[0].Utime)
3317         TimeValAdd(&ru1[0].Stime,ru2[0].Stime)
3318         TimeValAdd(&ru1[1].Utime,ru2[1].Utime)
3319         TimeValAdd(&ru1[1].Stime,ru2[1].Stime)
3320         return ru1
3321     }
3322     */
3323     // < name="rusage">Resource Usage</a>
3324     func Srusagef(fmtSpec string, argv []string, ru [2]aRusage)(string){
3325         // ru[0] self, ru[1] children
3326         u := TimeValAdd(ru[0].Utime,ru[1].Utime)
3327         s := TimeValAdd(ru[0].Stime,ru[1].Stime)
3328         /u := int64(u.Usec/1000000000 + int64(u.Usec) * 1000
3329         /u := ut; // in nano sec
3330         su := st; // in nano sec
3331         tu := uu + su
3332         ret := fmt.Sprintf("%v/um", abbtime(tu))
3333         ret += fmt.Sprintf("%v/sec", abbtme(uu))
3334         ret += fmt.Sprintf("%v/sys", abbtme(su))
3335         return ret
3336     }
3337     func Rusagef(fmtSpec string, argv []string, ru [2]aRusage)(string){
3338         at := TimeValAdd(ru[0].Utime,ru[1].Utime)
3339         st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3340         //fmt.Printf("%d.%06ds/u ",st.Sec,st.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
3341         fmt.Printf("%d.%06ds/u ",st.Sec,st.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
3342         fmt.Printf("%d.%06ds/u ",ut/1000000000,(ut/1000)*1000000);
3343         fmt.Printf("%d.%06ds/s ",st/1000000000,(st/1000)*1000000);
3344         return ""
3345     }
3346     func Getrusagev()([2]aRusage){
3347         var ruv = [2]aRusage{}
3348         aGetrusage(&RUSAGE_SELF, &ruv[0])
3349         aGetrusage(&RUSAGE_CHILDREN, &ruv[1])
3350         return ruv
3351     }
3352     func (gshctx *GshContext)xTime(argv[]string)(bool){
3353         if 2 < len(argv){
3354             gshctx.LastRusage = aRusage{}
3355             ru := aRusage()
3356             fin := gshctx.gshelv(argv[1:])
3357             ruusagev1 := Getrusagev()
3358             showRusage(argv[1], argv, gshctx.LastRusage)
3359             ruusage := RusageSub(ruusagev1,ruusagev2)
3360             showRusage("self", argv, ruusagev1)
3361             showRusage("child", argv, ruusagev2)
3362             return false
3363         }
3364     }
3365     func (gshctx *GshContext)xJobs(argv[]string){
3366         fmt.Println("# Jobs\n",len(gshctx.BackGroundJobs))
3367         for ji, pid := range gshctx.BackGroundJobs {
3368             //wstat := syscall.WaitStatus(0)
3369             //usage := aRusage {}
3370             //wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&usage);
3371             //wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&usage);
3372             wpid := pid.Pid();
3373             err := errors.New("stab_NoError");
3374             if err != nil {
3375                 fmt.Printf("--E-- %%d [%d] (%v)\n",ji,wpid,err)
3376             } else {
3377                 fmt.Printf("%%d[%d]\n",ji,wpid)
3378                 showRusage("child", argv, &usage)
3379             }
3380         }
3381     }
3382     func (gsh*gshContext)inBackground(argv[]string)(bool){
3383         if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n",argv) }
3384         gsh.Background = true // set background option
3385         xf1 := false
3386         xf1 = !gshelv(argv)
3387         gsh.Background = false
3388         return xf1
3389     }
3390 
```

```

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

```

```

3564 // scanf [format] [input]
3565 func scan(str string)(strv[]string{
3566     strv = strings.Split(str,",")
3567     return strv
3568 }
3569 func scanUntil(src,end string)(rstr string,leng int){
3570     idx := strings.Index(src,end)
3571     if 0 <= idx {
3572         rstr = src[0:idx]
3573         return rstr,idx+leng(end)
3574     }
3575     return src,0
3576 }
3577
3578 // -Bn -- scan base-name part only // can be in some %fmt, for sed rewriting
3579 func (gsh*GshContext)prinVcl(fmts string,vstr string,optv[]string){
3580     />int,err := strconv.Atoi(vstr)
3581     var ival int64 = 0
3582     n := 0
3583     err := error(nil)
3584     if strBegins(vstr,"-") {
3585         vx, _ := strconv.Atoi(vstr[1:])
3586         if vx < len(gsh.iValues) {
3587             vstr = gsh.iValues[vx]
3588         }else{
3589         }
3590     } // should use Eval()
3591     if strBegins(vstr,"0x") {
3592         n,err = fmt.Sscanf(vstr[2],"%x",&ival)
3593     }else{
3594         n,err = fmt.Sscanf(vstr,"%d",&ival)
3595     }
3596     //fmt.Printf("--D-- n=%d err=(%v) (%s)-v\n",n,err,vstr, ival)
3597
3598     if n == 1 && err == nil {
3599         //fmt.Printf("--D-- format(%v) ival(%v)\n",fmts,ival)
3600         fmt.Printf("%"+fmts,ival)
3601     }else{
3602         if isn("-bn",optv){
3603             fmt.Printf("%"+fmts,filepath.Base(vstr))
3604         }else{
3605             fmt.Printf("%"+fmts,vstr)
3606         }
3607     }
3608 }
3609 func (gsh*GshContext)prinfv(fmts,div string,argv[]string,optv[]string,list[]string{
3610     />int.Printf("(#d)",len(list))
3611     />curfmt := "v"
3612     outlen := 0
3613     curfmt := gsh.iFormat
3614
3615     if 0 < len(fmts) {
3616         for xi := 0; xi < len(fmts); xi++ {
3617             fch := fmts[xi]
3618             if fch == 'v' {
3619                 if xi+1 < len(fmts) {
3620                     curfmt = string(fmts[xi+1])
3621                     gsh.iFormat = curfmt
3622                     xi += 1
3623                     if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3624                         vals,leng := eval(fmts[xi+2:],")")
3625                         //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,leng)
3626                         gsh.printVal(curfmt,vals,optv)
3627                         xi += 2+leng-1
3628                         outlen += 1
3629                     }
3630                     continue
3631                 }
3632                 if fch == '-' {
3633                     hi,leng := scanInt(fmts[xi+1:])
3634                     if 0 < leng {
3635                         if hi < len(gsh.iValues) {
3636                             gsh.printVal(curfmt,gsh.iValues[hi],optv)
3637                             outlen += 1 // should be the real length
3638                         }else{
3639                             fmt.Printf("((out-range))")
3640                         }
3641                         xi += leng
3642                         continue;
3643                     }
3644                 }
3645                 fmt.Printf("%c",fch)
3646                 outlen += 1
3647             }else{
3648                 //fmt.Printf("--D-- print (%s)\n")
3649                 for i,v := range list {
3650                     if 0 < i {
3651                         fmt.Printf(div)
3652                     }
3653                     gsh.printVal(curfmt,v,optv)
3654                     outlen += 1
3655                 }
3656             }
3657             if 0 < outlen {
3658                 fmt.Printf("\n")
3659             }
3660         }
3661     }
3662 }
3663 func (gsh*GshContext)Scavn(argv[]string){
3664     />int.Printf("--D-- Scanv(%v)\n",argv)
3665     if len(argv) == 1 {
3666         return
3667     }
3668     argv = argv[1:]
3669     fmts = ""
3670     if strBegins(argv[0],"-F") {
3671         fmts = argv[0]
3672         gsh.iDelimiter = fmts
3673         argv = argv[1:]
3674     }
3675     input := strings.Join(argv, " ")
3676     if fmts == "" { // simple decomposition
3677         v := scanv(input)
3678         gsh.iValues = v
3679         //fmt.Printf("%v\n",strings.Join(v,""))
3680     }else{
3681         v := make([]string,0)
3682         n,err := fmt.Sscanf(input,fmts,&v[0],&v[1],&v[2],&v[3])
3683         fmt.Printf("--D-- Scanf ->(%v) n=%d err=(%v)\n",v,n,err)
3684         gsh.iValues = v
3685     }
3686 }
3687 func (gsh*GshContext)Printv(argv[]string{
3688     if false { //%%U
3689         fmt.Printf("%v\n",strings.Join(argv[1:], " "))
3690     }
3691 }
3692 //fmt.Printf("--D-- Printv(%v)\n",argv)
3693 //fmt.Printf("--D-- Printv(%v)\n",strings.Join(gsh.iValues,""))
3694 div := gsh.iDelimiter
3695 fmts := ""
3696 argv = argv[1:]
3697 if 0 < len(argv) {
3698     if strBegins(argv[0],"-F") {
3699         div = argv[1][2:]
3700         argv = argv[1:]
3701     }
3702 }
3703
3704 optv := []string{
3705     for v := range argv {
3706         if strBegins(v,"-"){
3707             optv = append(optv,v)
3708             argv = argv[1:]
3709         }else{
3710             break;
3711         }
3712     }
3713     if 0 < len(argv) {
3714         fmts = strings.Join(argv, " ")
3715     }
3716     gsh.printfv(fmts,div,argv,optv,gsh.iValues)
3717 }
3718 func (gsh*GshContext)Basename(argv[]string){
3719     for i,v := range gsh.iValues {
3720         gsh.iValues[i] = filepath.Base(v)
3721     }
3722 }
3723 func (gsh*GshContext)Sortv(argv[]string){
3724     sv := gsh.iValues
3725     sort.slice(sv , func(i,j int) bool {

```

```

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

```

```

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

```

```

4050         nil,
4051     }
4052 //fmt.Println("--I-- getline=&s // *\n",gsh_getlinev[0],gshCtx.GetLine)
4053 proc,err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&PA)
4054     if err != nil {
4055         fmt.Printf("--F-- getline process error (%v)\n",err)
4056         // for ; ; {
4057         return "exit (getline program failed)"
4058     }
4059 //stat, err := proc.Wait()
4060 proc.Wait()
4061 buff := make([]byte,LINESIZE)
4062 count,err := tty.Read(buff)
4063 //_,err = tty.Read(buff)
4064 //fmt.Println("--D-- getline (%d)\n",count)
4065 if err != nil {
4066     if !(count == 0) && err.String() == "EOF" {
4067         fmt.Printf("--E-- getline error (%s)\n",err)
4068     }
4069 }else{
4070     fmt.Printf("--I-- getline OK \\'s\\n",buff)
4071 }
4072 tty.Close()
4073 gline := string(buff[0:count])
4074 return gline
4075 }
4076 else{
4077 }
4078 // if isatty {
4079 //    fmt.Printf("!\#d",hix)
4080 //    fmt.Println(PROMPT)
4081 //}
4082 reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4083 line,_ := reader.ReadLine()
4084 return string(line)
4085 }
4086 }
4087 /**
4088 ** begin ===== getline
4089 */
4090 * getline.c
4091 * 2020-0819 extracted from dog.c
4092 * getline.go
4093 * 2020-0822 ported to Go
4094 */
4095 /*
4096 package main // getline main
4097 import (
4098     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
4099     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
4100     "os" // <a href="https://golang.org/pkg/os/">os</a>
4101     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
4102     "/bytes" // <a href="https://golang.org/pkg/bytes/">bytes</a>
4103     "/os/exec" // <a href="https://golang.org/pkg/os/">os</a>
4104 )
4105 */
4106
4107 // C language compatibility functions
4108 var errno = 0
4109 var stdin *os.File = os.Stdin
4110 var stdout *os.File = os.Stdout
4111 var stderr *os.File = os.Stderr
4112 var EOF = -1
4113 var NULL = 0
4114 type FILE os.File
4115 type Stat struct{[]byte}
4116 var NULL FP os.File = nil
4117 var NULLP = 0
4118 //var LINESIZE = 1024
4119
4120 func system(cmdstr string)(int){
4121     // /proc/self/cAttr {
4122     PA := os.ProcAttr{
4123         "", // the starting directory
4124         os.Environ(),
4125         //|uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd(),
4126         []*os.File(os.Stdin,os.Stdout,os.Stderr),
4127         nil,
4128     }
4129     argv := strings.Split(cmdstr," ")
4130     //pid,err := syscall.ForkExec(argv[0],argv,&PA)
4131     proc,err := os.StartProcess(argv[0],argv,&PA);
4132     if(err != nil){
4133         //fmt.Println("--E-- system(%v)\n(%v)",cmdstr,err);
4134         return -1;
4135     }
4136     pstat_r := proc.Wait();
4137     Pid := pstat.Pid();
4138     if( err != nil ){
4139         fmt.Println("--E-- pid=%v syscall(%v) err(%v)\n",pid,cmdstr,err)
4140     }
4141     //syscall.Wait4(pid,nil,0,nil)
4142     //fmt.Println("==== pid=%d exit=%v stat=%v\n",pid,pstat.Exited(),pstat.ExitCode());
4143
4144     /*
4145     argv := strings.Split(cmdstr," ")
4146     fmt.Fprintf(os.Stderr,"--I-- system(%v)\n",argv)
4147     //cmd := exec.Command(argv[0];...)
4148     cmd := exec.Command(argv[0],argv[1],argv[2])
4149     cmd.Stdin = strings.NewReader("output of system")
4150     var serr bytes.Buffer
4151     cmd.Stdout = &serr
4152     var serr bytes.Buffer
4153     cmd.Stderr = &serr
4154     err := cmd.Run()
4155     if( err != nil ){
4156         fmt.Fprintf(os.Stderr,"--E-- system(%v)err(%v)\n",argv,err)
4157         fmt.Println("ERR:%s",serr.String())
4158     }else{
4159         fmt.Println("%s",out.String())
4160     }
4161     */
4162     return 0
4163 }
4164 func atoi(str string)(ret int){
4165     ret,err := fmt.Sscanf(str,"%d",ret)
4166     if err == nil {
4167         return ret
4168     }else{
4169         // should set errno
4170         return 0
4171     }
4172 }
4173 func getenv(name string)(string){
4174     val,got := os.LookupEnv(name)
4175     if got {
4176         return val
4177     }else{
4178         return "?"
4179     }
4180 }
4181 func strcpy(dst StrBuff, src string){
4182     var i int
4183     srcb := []byte(src)
4184     for i = 0; i < len(src) && srcb[i] != 0; i++ {
4185         dst[i] = srcb[i]
4186     }
4187     dst[i] = 0
4188 }
4189 func xstrcpy(dst StrBuff, src StrBuff){
4190     dst = src
4191 }
4192 func strcat(dst StrBuff, src StrBuff){
4193     dst = append(dst,src...)
4194 }
4195 func strdup(str StrBuff)(string){
4196     return string(str[:strlen(str)])
4197 }
4198 func strlen(str string)(int){
4199     return len(str)
4200 }
4201 func strlen(str StrBuff)(int){
4202     var i int
4203     for i = 0; i < len(str) && str[i] != 0; i++ {
4204         }
4205     return i
4206 }
4207 func sizeof(data StrBuff)(int){
4208     return len(data)
4209 }
4210 func isatty(fd int)(ret int){
4211     return 1

```

```

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

```

```

4374         case "-b": builtin = true
4375         case "-d": debug = true
4376         case "-s": silent = true
4377         case "-v": info = true
4378     }
4379     opt = argv[1]
4380     argv = argv[2:]
4381 }
4382
4383 dicName := ""
4384 dicURL := ""
4385 if 0 < len(argv) {
4386     arg0 := argv[0]
4387     dicName = arg0
4388     argv = argv[1:]
4389 }
4390 if 0 < len(argv) {
4391     dicURL = argv[0]
4392     argv = argv[1:]
4393 }
4394 if false {
4395     fprintf(stderr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
4396 }
4397 if cmd == DIC_COM_LOAD {
4398     //dicType := "load"
4399     dicBody := ""
4400     if !builtin && dicName != "" && dicURL == "" {
4401         f,err := os.Open(dicName)
4402         if err != nil {
4403             dicURL = dicName
4404         } else {
4405             f,err = os.Open(dicName+".html")
4406             if err == nil {
4407                 dicURL = dicName+".html"
4408             } else {
4409                 f,err = os.Open("gshdic-"+dicName+".html")
4410                 if err == nil {
4411                     dicURL = "gshdic-"+dicName+".html"
4412                 }
4413             }
4414         }
4415     }
4416     if err == nil {
4417         var buf := make([]byte,128*1024)
4418         count,err := f.Read(buf)
4419         f.Close()
4420         if info {
4421             fprintf(stderr,"--Id-- ReadDic(%v,%v)\n",count,err)
4422         }
4423         dicBody = string(buf[0:count])
4424     }
4425     if dicBody == "" {
4426         switch arg {
4427             default:
4428                 dicName = "WorldDic"
4429                 dicURL = WorldDic
4430                 if info {
4431                     fprintf(stderr,"--Id-- default dictionary \"%v\"\n",
4432                           dicName);
4433             }
4434             case "wnn":
4435                 dicName = "WnnDic"
4436                 dicURL = WnnDic
4437             case "sURL":
4438                 dicName = "SumomoDic"
4439                 dicURL = sumomoDic
4440             case "sijimi":
4441                 dicName = "SijimiDic"
4442                 dicURL = SijimiDic
4443             case "x1l":
4444                 dicName = "JKLJaDic"
4445                 dicURL = JA_JKLDic
4446         }
4447     }
4448     if debug {
4449         fprintf(stderr,"--Id-- %v URL=%v\n",dicName,dicURL);
4450     }
4451     dicv := strings.Split(dicURL,",")
4452     if debug {
4453         fprintf(stderr,"--Id-- %v encoded data...\n",dicName)
4454         fprintf(stderr,"%v\n",dicv[0])
4455         fprintf(stderr,"%v\n",dicv[1])
4456         fprintf(stderr,"%v\n",dicv[2])
4457     }
4458     body_ := base64.StdEncoding.DecodeString(dicv[1])
4459     dicBody = string(body_)
4460 }
4461 if info {
4462     fmt.Printf("--Id-- %v %v\n",dicName,dicURL)
4463     fmt.Printf("%s\n",dicBody)
4464 }
4465 if debug {
4466     fprintf(stderr,"--Id-- dicName %v text...\n",dicName)
4467     fprintf(stderr,"%v\n",string(dicBody))
4468 }
4469 envt := strings.Split(dicBody,"\n");
4470 if info {
4471     fprintf(stderr,"--Id-- %v scan...\n",dicName);
4472 }
4473 var added int = 0
4474 var dup int = 0
4475 for i,v := range envt {
4476     var pat string
4477     var out string
4478     fmt.Sscanf(v,"%s %s",&pat,&out)
4479     if len(pat) <= 0 {
4480         if else{
4481             if 0 <= isinDic(pat) {
4482                 dup += 1
4483                 continue
4484             }
4485             romkan := RomKana(dicName,pat,out,0)
4486             dicName += 1
4487             added += 1
4488             Romkan = append(Romkan,RomKana(dicName,pat,out,0))
4489             if debug {
4490                 fmt.Printf("[%v]:[%2v]-%v [%2v]\n",
4491                   i,len(pat),pat,len(out),out)
4492             }
4493         }
4494     }
4495     if !silent {
4496         url := dicURL
4497         if strbegins(url,"data:") {
4498             url = "builtin"
4499         }
4500         fprintf(stderr,"--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4501               dicName,added,dup,len(Romkan),url);
4502     }
4503     // should sort by pattern length for conclete match, for performance
4504     if debug {
4505         arg = "" // search pattern
4506         dump = true
4507     }
4508 }
4509 if cmd == DIC_COM_DUMP || dump {
4510     fprintf(stderr,"--Id-- %v dump... %v entries:\n",dicName,len(Romkan));
4511     var match = 0
4512     for i := 0; i < len(Romkan); i++ {
4513         dic := Romkan[i].dic
4514         pat := Romkan[i].pat
4515         out := Romkan[i].out
4516         if arg == "" || 0 <= strings.Index(pat,arg)|| 0 <= strings.Index(out,arg) {
4517             fmt.Printf("\n\\%v [%2v]-%v [%2v]\n",
4518               i,dic,len(pat),pat,len(out),out)
4519             match += 1
4520         }
4521     }
4522     fprintf(stderr,"--Id-- %v matched %v / %v entries:\n",arg,match,len(Romkan));
4523 }
4524 }
4525 func loadDefaultDic(dic int){
4526     if 0 < len(Romkan) {
4527         return
4528     }
4529     //fprintf(stderr,"r\n")
4530     xDic([]string{"dic",DIC_COM_LOAD});
4531
4532     var info = false
4533     if info {
4534         fprintf(stderr,"--Id-- Congururations!! WorldDic is now activated.\r\n")
4535         fprintf(stderr,"--Id-- enter \"dic\" command for help.\r\n")

```

```

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

```

```

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

```

```

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

```

```

5022 // record a Keyin(ch) Event
5023 {
5024     now := time.Now()
5025     if 0 < len(Events) {
5026         last := Events[len(Events)-1]
5027         dura := int64(now.Sub(last.When))
5028         Events = append(Events,Event{last.When,Ev_IDLE,dura,last.CmdIndex})
5029     }
5030     Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5031 }
5032 return ch;
5033 case ch == <- evQ:
5034     if( ch == EV_MODE ){
5035         recvKeyEvent()
5036     }else{
5037         return ch;
5038     }
5039 }
5040 }
5041 func putKeyEvent(event int, evarg int){
5042     evQ <- event;
5043 }
5044 func recvKeyEvent(){
5045     ch := <- evQ;
5046     if( ch != 0 ){
5047         romkanmode = true
5048     }else{
5049         romkanmode = false
5050     }
5051 }
5052 }
5053 var AtConsoleLineTop = true
5054 var TtyMaxCol = 72 // to be obtained by ioctl?
5055 var EscTimeout = (100*1000)
5056 var (
5057     MODE_VicMode bool // vi compatible command mode
5058     MODE_Shadow bool // shown translation mode, the mode to be retained
5059     MODE_CapsLock bool // show caps lock translation
5060     MODE_LockWord bool // force lower-case character lock
5061     MODE_Vinsert int // visible insert mode, should be like "I" icon in X Window
5062     MODE_Vitrace bool // output newline before translation
5063 )
5064 type IInput struct {
5065     lno int
5066     lastlno int
5067     pch []int // input queue
5068     prompt string
5069     line string
5070     right string
5071     inMode bool
5072     pinMode bool
5073     waitingMeta string // waiting meta character
5074     LastCmd string
5075 }
5076 func (iin*IInput)Gett(timeoutUs int)(int){
5077     ch1 := EOF
5078     ch2 := EOF
5079     ch3 := EOF
5080     if( 0 < len(iin.pch) ){ // deq
5081         ch1 = iin.pch[0]
5082         iin.pch = iin.pch[1];
5083     }else{
5084         ch1 = fgetTimeout(stdin,timeoutUs);
5085     }
5086     if( ch1 == 033 ){ // escape sequence
5087         ch1 = fgetTimeout(stdin,EscTimeout);
5088         if(ch2 == EV_TIMEOUT ){
5089             ch3 = fgetTimeout(stdin,EscTimeout);
5090             if( ch3 == EV_TIMEOUT ){
5091                 in.pch = append(iin.pch,ch2) // enq
5092             }else{
5093                 switch( ch2 ){
5094                     default:
5095                         iin.pch = append(iin.pch,ch2) // enq
5096                         iin.pch = append(iin.pch,ch3) // enq
5097                     case '[':
5098                         switch( ch3 ){
5099                             case 'A': ch1 = GO_UP; // ^
5100                             case 'B': ch1 = GO_DOWN; // v
5101                             case 'C': ch1 = GO_RIGHT; // >
5102                             case 'D': ch1 = GO_LEFT; // <
5103                             case 'H':
5104                             case 'M':
5105                             case 'P':
5106                             case 'R':
5107                         }
5108                         ch4 := fgetTimeout(stdin,EscTimeout);
5109                         if( ch4 == '-' ){
5110                             //ffprintf(stderr,"%02X %02X %02X\n",ch1,ch2,ch3,ch4);
5111                             ch1 = DEL_RIGHT
5112                         }
5113                     }
5114                 }
5115             }
5116         }
5117     }
5118 }
5119 return ch1
5120 }
5121 }
5122 }
5123 return ch
5124 }
5125 func (iin*IInput)clearline(){
5126     var i int
5127     fprintf(stderr,"r");
5128     // should be ANSI ESC sequence
5129     for i = 0; i < TtyMaxCol; i++ // to the max. position in this input action
5130         fputc(' ',os.Stdout);
5131     }
5132     fprintf(stderr,"r");
5133 }
5134 func (iin*IInput)Redraw(){
5135     redraw(iin.lno,iin.in,iin.line,iin.right)
5136 }
5137 }
5138 func redrawiin *Input,lno int,line string,right string{
5139     inMeta := false
5140     showMode := ""
5141     showMeta := "" // visible Meta mode on the cursor position
5142     showInk := fmt.Sprintf("%d! ",lno)
5143     InsertMark := "" // in visible insert mode
5144
5145     if MODE_VicMode {
5146     }else{
5147         if 0 < len(iin.right) {
5148             InsertMark = "+"
5149         }
5150
5151         if( 0 < len(iin.waitingMeta) ){
5152             inMeta = true
5153             if iin.waitingMeta[0] != 033 {
5154                 showMeta = iin.waitingMeta
5155             }
5156         }
5157         if( romkanmode ){
5158             romkanmark = "*";
5159         }else{
5160             romkanmark = "";
5161         }
5162         if MODE_ShowMode {
5163             romkan := "-";
5164             inmeta := "-";
5165             inveri := "";
5166             if MODE_CapsLock {
5167                 inmeta = "A";
5168             }
5169             if MODE_LockWord {
5170                 inmeta = "a";
5171             }
5172             if MODE_Vitrace {
5173                 inveri = "v";
5174             }
5175             if MODE_VicMode {
5176                 inveri = "=";
5177             }
5178             if romkanmode {
5179                 romkan = "1343\201\202";
5180                 if MODE_CapsLock {
5181                     inmeta = "R";
5182                 }else{
5183                     inmeta = "r";
5184                 }
5185             }
5186         }
5187     }
5188 }
```

```

5184     }
5185     if(inMeta) {
5186         inmeta = "\\"+
5187     }
5188     showMode = "["+romkan+inmeta+inveri+"]";
5189 }
5190 Pre := "\r" + showMode + showLino
5191 Output += Pre
5192 Left := ""
5193 Right := ""
5194 if(romkanmode) {
5195     Left = convs(line)
5196     Right = InsertMark+convs(right)
5197 }
5198 Left = line
5199 Right = InsertMark+right
5200 }
5201 Output = Pre+left
5202 if(MODE_ViTrace) {
5203     Output += in.LastCmd
5204 }
5205 Output += showMetaRight
5206 for(len(Output) < TtyMaxCol) { // to the max. position that may be dirty
5207     Output += " "
5208     // should be ANSI ESC sequence
5209     // not necessary just after newline
5210 }
5211 Output += PreLeftshowMeta // to set the cursor to the current input position
5212 fprintf(stderr,"$s",Output)
5213 if(MODE_ViTrace) {
5214     if(0 < len(in.LastCmd)) {
5215         in.LastCmd = ""
5216         fprintf(stderr,"\r\n")
5217     }
5218 }
5219 AtconsoleLinePop = false
5220 //fmt.Printf("(Redraw(%v))\n",len(line),len(right));
5221 } // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
5222 func delHeadChar(str string)(line string,head string){
5223     clen := len(str)
5224     head = string(str[:clen])
5225     return str[clen:],head
5226 }
5227 func delta1Char(str string)(rline string, last string){
5228     var i = 0
5229     var clen = 0
5230     for {
5231         ,siz := utf8.DecodeRune([]byte(str)[i:])
5232         if siz <= 0 { break }
5233         clen = siz
5234         i += siz
5235     }
5236     last = str[len(str)-clen:]
5237     return str[0:len(str)-clen],last
5238 }
5239 // 3* for output and history
5240 // 4* for keylog?
5241 // <a name="getline">Command Line Editor</a>
5242 func xgetline(iin int, prevline string, gsh*GshContext)(string){
5243     var iin Input
5244     iin.lastline = iin.line
5245     iin.lno = iin.line
5246     CmdIndex = len(gsh.CommandHistory)
5247     if(isatty(0) == 0) {
5248         if(sfgets(&iin.line,LINESIZE,stdin) == NULL) {
5249             iin.line = "exit\n"
5250         } else{
5251         }
5252         return iin.line
5253     }
5254     if(true) {
5255         //var pts string;
5256         //pts = ptsname(0);
5257         //if(ttyname(0));
5258         //fprintf(stderr,"-pts[0] = %s\n",pts?pts:"");
5259     }
5260     if(false) {
5261         fprintf(stderr,"! ");
5262         ffdiscard(stdin,LINESIZE,stdin);
5263         sfgets(&iin.line,LINESIZE,stdin);
5264         return iin.line
5265     }
5266     if(!OnWindows) { system("/bin/stty -echo -icanon"); }
5267     xline := iin.xgetline(prevline,gsh)
5268     if(!OnWindows) {system("/bin/stty echo sane"); }
5269     return xline
5270 }
5271 func (iin*Input)Translate(cmdch int){
5272     romkanmode = !romkanmode;
5273     if(MODE_ViTrace) {
5274         fprintf(stderr,"%v\r\n",string(cmdch));
5275     } else{
5276         if(cmdch == 'J') {
5277             fprintf(stderr,"J\r\n");
5278         }
5279         iin.inJMode = true
5280     }
5281     iin.Redraw();
5282     loadDefaultDic(cmdch);
5283     iin.Redraw();
5284 }
5285 func (iin*Input)Replace(cmdch int){
5286     iin.LastCmd = fmt.Sprintf("\%v",string(cmdch))
5287     iin.Redraw();
5288     loadDefaultDic(cmdch);
5289     dst := convs(iin.line+iin.right);
5290     iin.line = dst
5291     iin.right = ""
5292     if(iin.line[iin.line-1] == 'I'){
5293         fprintf(stderr,"I\r\n");
5294         iin.inJMode = true
5295     }
5296     iin.Redraw();
5297 }
5298 func (iin*Input)Replace(cmdch int){
5299     iin.Redraw();
5300     loadDefaultDic(cmdch);
5301     dst := convs(iin.line+iin.right);
5302     iin.line = dst
5303     iin.right = ""
5304     if(iin.line[iin.line-1] == 'I'){
5305         fprintf(stderr,"I\r\n");
5306         iin.inJMode = true
5307     }
5308     iin.Redraw();
5309 }
5310 // aa 12 ala1
5311 func isAlpha(ch rune)(bool){
5312     if('a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5313         return true
5314     }
5315     return false
5316 }
5317 }
5318 // 0.2.8 2020-09-01 created
5319 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
5320 func (iin*Input)GetTOFW(){
5321     str := iin.line
5322     i := len(str)
5323     if(i < 0) {
5324         return
5325     }
5326     /i0 := i
5327     i -= 1
5328     lastSize := 0
5329     var lastRune rune
5330     var found = -1
5331     for 0 < i { // skip preamble spaces
5332         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5333         if !isAlnum(lastRune) { // character, type, or string to be searched
5334             i -= lastSize
5335             continue
5336         }
5337     }
5338     break
5339 }
5340 for 0 < i {
5341     lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5342     if lastsize <= 0 { continue } // not the character top
5343     if !isAlnum(lastRune) { // character, type, or string to be searched
5344         found = i
5345     }
5346     break
5347 }
5348 }
5349 }
5350 }
5351 }
5352 }
5353 }
5354 }
5355 }
5356 }
5357 }
5358 }
5359 }
5360 }
5361 }
5362 }
5363 }
5364 }
5365 }
5366 }
5367 }
5368 }
5369 }
5370 }
5371 }
5372 }
5373 }
5374 }
5375 }
5376 }
5377 }
5378 }
5379 }
5380 }
5381 }
5382 }
5383 }
5384 }
5385 }
5386 }
5387 }
5388 }
5389 }
5390 }
5391 }
5392 }
5393 }
5394 }
5395 }
5396 }
5397 }
5398 }
5399 }
5400 }
5401 }
5402 }
5403 }
5404 }
5405 }
5406 }
5407 }
5408 }
5409 }
5410 }
5411 }
5412 }
5413 }
5414 }
5415 }
5416 }
5417 }
5418 }
5419 }
5420 }
5421 }
5422 }
5423 }
5424 }
5425 }
5426 }
5427 }
5428 }
5429 }
5430 }
5431 }
5432 }
5433 }
5434 }
5435 }
5436 }
5437 }
5438 }
5439 }
5440 }
5441 }
5442 }
5443 }
5444 }
5445 }
5446 }
5447 }
5448 }
5449 }
5450 }
5451 }
5452 }
5453 }
5454 }
5455 }
5456 }
5457 }
5458 }
5459 }
5460 }
5461 }
5462 }
5463 }
5464 }
5465 }
5466 }
5467 }
5468 }
5469 }
5470 }
5471 }
5472 }
5473 }
5474 }
5475 }
5476 }
5477 }
5478 }
5479 }
5480 }
5481 }
5482 }
5483 }
5484 }
5485 }
5486 }
5487 }
5488 }
5489 }
5490 }
5491 }
5492 }
5493 }
5494 }
5495 }
5496 }
5497 }
5498 }
5499 }
5500 }
5501 }
5502 }
5503 }
5504 }
5505 }
5506 }
5507 }
5508 }
5509 }
5510 }
5511 }
5512 }
5513 }
5514 }
5515 }
5516 }
5517 }
5518 }
5519 }
5520 }
5521 }
5522 }
5523 }
5524 }
5525 }
5526 }
5527 }
5528 }
5529 }
5530 }
5531 }
5532 }
5533 }
5534 }
5535 }
5536 }
5537 }
5538 }
5539 }
5540 }
5541 }
5542 }
5543 }
5544 }
5545 }
5546 }
5547 }
5548 }
5549 }
5550 }
5551 }
5552 }
5553 }
5554 }
5555 }
5556 }
5557 }
5558 }
5559 }
5560 }
5561 }
5562 }
5563 }
5564 }
5565 }
5566 }
5567 }
5568 }
5569 }
5570 }
5571 }
5572 }
5573 }
5574 }
5575 }
5576 }
5577 }
5578 }
5579 }
5580 }
5581 }
5582 }
5583 }
5584 }
5585 }
5586 }
5587 }
5588 }
5589 }
5590 }
5591 }
5592 }
5593 }
5594 }
5595 }
5596 }
5597 }
5598 }
5599 }
5599 }
```

```

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

```

5508     i += z
5509   }
5510   if 0 <= found {
5511     iin.line = iin.line + right[0:found]
5512     iin.right = iin.right[found:]
5513     return true
5514   }else{
5515     return false
5516   }
5517 }
5518 // 0.2.8 2020-0902 created
5519 func (lin:Input)searchBackward(pat string)(bool){
5520   line := iin.line
5521   found := -1
5522   i := len(line)-1
5523   for i := i; 0 <= i; i-- {
5524     _,z := util.DecodeRuneInString(line[i:])
5525     if z == 0 {
5526       continue
5527     }
5528     //printf(stderr,"-- %v\n",pat,line[i:])
5529     if strBegins(line[i:],pat) {
5530       found = i
5531       break
5532     }
5533   }
5534   //printf(stderr,"--%d\n",found)
5535   if 0 <= found {
5536     iin.line = line[0:found] + iin.right
5537     iin.right = line[0:found]
5538     return true
5539   }else{
5540     return false
5541   }
5542 }
5543 // 0.2.8 2020-0902 created
5544 // search from top, end, or current position
5545 func (gsh:GshContext)SearchHistory(pat string, forw bool)(bool,string){
5546   if forw {
5547     for _v := range gsh.CommandHistory {
5548       if 0 <= strings.Index(_v.Cmdline,pattern) {
5549         //printf(stderr,"-->%v\n",_v.Cmdline)
5550         return true,_v.Cmdline
5551     }
5552   }else{
5553     hlen := len(gsh.CommandHistory)
5554     for i := hlen-1; 0 <= i; i- {
5555       v := gsh.CommandHistory[i]
5556       if 0 <= strings.Index(v.Cmdline,pattern) {
5557         //printf(stderr,"--De-- found %v\n",v.Cmdline)
5558         return true,v.Cmdline
5559       }
5560     }
5561   }
5562   //printf(stderr,"n--De-- not-found(%v)\n",pattern)
5563   return false,(Not Found in History)
5564 }
5565 // 0.2.8 2020-0902 created
5566 func (lin:Input)gotFORWSTR(pat string,gsh:GshContext){
5567   found := false
5568   if 0 < len(iin.right) {
5569     found = iin.SearchForward(pattern)
5570   }
5571   if !found {
5572     found, line := gsh.SearchHistory(pattern,true)
5573     if found {
5574       iin.line = line
5575       iin.right = ""
5576     }
5577   }
5578 }
5579 func (lin:Input)GotoBACKSTR(pat string, gsh:GshContext){
5580   found := false
5581   if 0 < len(iin.line) {
5582     found = iin.SearchBackward(pattern)
5583   }
5584   if !found {
5585     found, line := gsh.SearchHistory(pattern,false)
5586     if found {
5587       iin.line = line
5588       iin.right = ""
5589     }
5590   }
5591 }
5592 func (lin:Input)getStringi(prompt string)(string){ // should be editable
5593   var clearLine := true
5594   fprintf(stderr,"%v",prompt)
5595   str := ""
5596   for {
5597     ch := iin.Getc(10*1000*1000)
5598     if ch == '\n' || ch == '\r' {
5599       break
5600     }
5601     sch := string(ch)
5602     str += sch
5603     fprintf(stderr,"%s",sch)
5604   }
5605   return str
5606 }
5607 // search pattern must be an array and selectable with '^N'/'P'
5608 var SearchPat : string
5609 var SearchBack = true
5610 var SearchForw = true
5611 func (lin:Input)getLine1(prevline string, gsh:GshContext)(string){
5612   var ch int;
5613   MODE_ShowMode = false
5614   MODE_VicMode = false
5615   iin.Redraw();
5616   first := true
5617   for cix := 0; ; cix++ {
5618     iin.pinMode = iin.inMode
5619     iin.inMode = false
5620
5621     ch := iin.Getc(1000*1000)
5622
5623     if ch == EV_TIMEOUT && first {
5624       first = false
5625       mode := 0
5626       if romkanemode {
5627         mode = 1
5628       }
5629       now := time.Now()
5630       Events = append(Events,Event(now,EV_MODE,int64(mode),CmdIndex))
5631     }
5632     if ch == 033 {
5633       MODE_ShowMode = true
5634       MODE_VicMode = !MODE_VicMode
5635       iin.Redraw();
5636       continue
5637     }
5638   if MODE_VicMode {
5639     switch ch {
5640     case '0': ch = GO_TOPL
5641     case '3': ch = GO_ENDL
5642     case 'b': ch = GO_TOPW
5643     case 'e': ch = GO_ENDW
5644     case 'w': ch = GO_NEXTW
5645     case 's': ch = GO_PAIRW
5646
5647     case 'j': ch = GO_DOWN
5648     case 'k': ch = GO_UP
5649     case 'h': ch = GO_LEFT
5650     case 'l': ch = GO_RIGHT
5651     case 'x': ch = OEL_RIGHT
5652     case 'a': MODE_VicMode = !MODE_VicMode
5653     ch = GO_RIGHT
5654     case 'i': MODE_VicMode = !MODE_VicMode
5655     iin.Redraw();
5656     continue
5657     case 'c':
5658       if iin.head == delHeadChar(iin.right)
5659       if len([]byte(head)) == 1 {
5660         ch = int(head[0])
5661         if 'a' <= ch && ch <= 'z' {
5662           ch = ch + 'A'-'a'
5663         }else{
5664           if 'A' <= ch && ch <= 'Z' {
5665             ch = ch + 'a'-'A'
5666           }
5667         }
5668       }
5669     }
5670   }
5671   iin.head := delHeadChar(iin.right)
5672   if len([]byte(head)) == 1 {
5673     ch = int(head[0])
5674     if 'a' <= ch && ch <= 'z' {
5675       ch = ch + 'A'-'a'
5676     }else{
5677       if 'A' <= ch && ch <= 'Z' {
5678         ch = ch + 'a'-'A'
5679       }
5680     }
5681   }
5682 }

```

```

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

```

```

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

```

5994     *dp = 0;
5995   }
5996
5997   if( 0 ){
5998     fprintf(stderr,"%n%d\n",int(strlen(line)));
5999   }
6000   if( lseek(3,0,0) == 0 ){
6001     if( romkammod ){
6002       var buf [8*124]byte;
6003       convert(line,buf);
6004       strcpy(line,buf);
6005     }
6006     write(3,line,strlen(line));
6007     ftruncate(3,lseek(3,0,SEEK_CUR));
6008     //fseek(3,0,SEEK_SET,offset=+d\n",int)lseek(3,0,SEEK_END));
6009     lseek(3,0,SEEK_SET);
6010     close(3);
6011   }else{
6012     fprintf(stderr,"r\nngoline: ");
6013     trans(line);
6014     //printf("%s\n",line);
6015     printf("\n");
6016   }
6017 */
6018 /**
6019  */
6020 //== end ===== getline
6021 //
6022 // _SUSERHOME/_gsh/
6023 //   gsh-rc.txt, or gsh-configure.txt
6024 //   gsh-history.txt
6025 //   gsh-aliases.txt // should be conditional?
6026 //
6027 func (gshctx *GshContext)gshSetupPhomedir()(bool) {
6028   homedir,found := userHomeDir()
6029   if !found {
6030     fmt.Println("--E-- You have no UserHomeDir\n")
6031     return true
6032   }
6033   gshhome := homedir + "/" + GSH_HOME
6034   err2 := os.Stat(gshhome)
6035   if err2 != nil {
6036     err1 := os.Mkdir(gshhome,0700)
6037     if err1 != nil {
6038       fmt.Printf("--E-- Could not Create %s (%s)\n",
6039         gshhome,err3)
6040       return true
6041     }
6042     fmt.Println("--I-- Created %s\n",gshhome)
6043   }
6044   gshctx.GshHomeDir = gshhome
6045   return false
6046 }
6047 func setupGshContext()(GshContext,bool){
6048   //gshctx := setupGsh();
6049   gshPA := os.Pwd();
6050   // the starting directory
6051   os.Environ(), // environ[]
6052   //[]uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd(),
6053   []*os.File(os.Stdin,os.Stdout,os.Stderr),
6054   nil, // OS specific
6055   cwd, := os.Getwd()
6056   gshctx := GshContext {
6057     cwd, // StartDir
6058     "", // GetLine
6059     []GchDirHistory { { cwd,time.Now(),0 } }, // ChdirHistory
6060     gshPA,
6061     []GCommandHistory{}, //Something for invocation?
6062     GCommandHistory{}, // CmdCurrent
6063     false,
6064     []os.ProcessState{}, //[]int{},
6065     abusage(),
6066     "", // GshHomeDir
6067     TtyId(),
6068     false,
6069     false,
6070     []PluginInfo{},
6071     []string{},
6072     "-",
6073     "v",
6074     ValueStack{},
6075     GServer{"/"}, // LastServer
6076     "tcp4:5555",
6077     cwd, // RWD
6078     CheckSum{},
6079   }
6080   err := gshctx.gshSetupHomedir()
6081   return gshctx, err
6082 }
6083 }
6084 func (gsh*GshContext)tgshellh(gline string)(bool){
6085   ghist := gsh.CmdCurrent
6086   ghist.WorkDir_ = os.Getwd()
6087   ghist.WorkDir_ = len(gsh.ChdirHistory)-1
6088   //"/"+os.Getenv("PWD")+"\n"+len(gsh.ChdirHistory)+"#d\n",len(gsh.ChdirHistory))
6089   ghist.StartAt = time.Now()
6090   rusagev1 := Getrusagev()
6091   gsh.CmdCurrent.Foundfile = []string{}
6092   fin := gsh.tgshellh(gline)
6093   rusagev2 := Getrusagev()
6094   ghist.Rusagev = append(gsh.Rusagev,rusagev1)
6095   ghist.Endat = time.Now()
6096   ghist.Cmedline = gline
6097   ghist.Foundfile = gsh.CmdCurrent.Foundfile
6098   /* record it but not show in list by default
6099   if len(gline) == 0 {
6100     continue
6101   }
6102   if gline == "hi" || gline == "history" { // don't record it
6103     continue
6104   }
6105   */
6106   gsh.CommandHistory = append(gsh.CommandHistory, ghist)
6107   return fin
6108 }
6109 //}
6110 // <a name="main">Main loop</a>
6111 func scroll(gshctxGiven *GshContext) (_ GshContext) {
6112   gshctxBuf,err0 := setupGshContext()
6113   if err0 {
6114     return gshctxBuf;
6115   }
6116   gshctx := *gshctxBuf
6117
6118   //fmt.Printf("--I-- GSH_HOME=%s\n",gshctx.GshHomeDir)
6119   //resmap()
6120
6121 /*
6122 if false {
6123   gsh_getlinev, with_execline :=
6124     which("PATH",[]string{"which","gsh-getline","-s"})
6125   if with_execline {
6126     gsh_getlinev[0] = toFullPath(gsh_getlinev[0])
6127     gshctx.Getline = toFullPath(gsh_getlinev[0])
6128   }else{
6129     fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
6130   }
6131 }
6132 */
6133
6134 ghist0 := gshctx.CmdCurrent // something special, or gshrc script, or permanent history
6135 gshctx.CommandHistory = append(gshctx.CommandHistory,ghist0)
6136
6137 prevline := ""
6138 skipping := false
6139 for hix := len(gshctx.CommandHistory); {
6140   gline := gshctx.getline(hix,skipping,prevline)
6141   if skipping {
6142     if strings.Index(gline,"fi") == 0 {
6143       fmt.Printf("fi\n");
6144       skipping = false;
6145     }else{
6146       //fmt.Printf("%s\n",gline);
6147     }
6148     continue
6149   }
6150   if strings.Index(gline,"if") == 0 {
6151     //fmt.Printf("--D-- if start: %s\n",gline);
6152     skipping = true;
6153     continue
6154   }
6155   if false {

```



```

6318 //</span>
6319 var sijiminic = "<span id="gsh-sijim-dic">
6320 <data:text/dic;base64,"+
6321 "PG1ldGsgt2hcnNldD0iVVRGLTgiPg08dGV4dGFyZWRgY29scz04MCByb3dzPTQwPgovL3z1"+
6322 "cglnU2h1lbKcc0lNRVxzG1jdg1vbmyExz9yXHNTxG1qaw1pXHMvLzxMjAyMCwODMw"+
6323 "CNpCeOB1pzaGkJ44GKcmppCeBmAptaqNjgb8khmJ44GqCmp1CeOBmOChQ4eXU44KF"+
6324 "amw44GICmtq2trbnAnjgOk2prazwJ44GICmpramtrAnjgYOK2tramJ44GCPmramps"+
6325 "Cm044GICmtq2prbnAnjgOk2prazwJ44GICmpramtrAnjgYOK2tramJ44GCPmramps"+
6326 "Cm044GICmtq2prbnAnjgOk2prazwJ44GICmpramtrAnjgYOK2tramJ44GCPmramps"+
6327 "SgpmCmtrveahm1lgsKa29rCmWA1wp1b3gJ5YCLCm5hbmFgqXVuaXgJNjIKhmFuYp1d5p1g9j"+
6328 "7y7hcmFxCeOeoB1+OC1oP0awthcmE5J5YqbcNnaWthcmE5J5YqbcjwvdGv4dGfYzWE+Cg"+
6329 "awthcmFxCeOeoB1+OC1oP0awthcmE5J5YqbcNnaWthcmE5J5YqbcjwvdGv4dGfYzWE+Cg"+
6330 //</span>
6331
6332 var JA_JKDdic = "<span id="gsh-ja-jk1-dic">
6333 <data:text/dic;base64,"+
6334 "Pg9ZXJscCU158U1FanRpYZptb3jZwRKOwKSxWmjAyGGoDE5KsheLV4pL1nhhG94SVRT"+
6335 "CmtpqprGtqaa2sa2p1OGs1ueV1ApqamtqamJ44GCCmtqAnjgYQRa2tgdnjhYYKamtg"+
6336 "amw44GICmtq2trbnAnjgOk2prazwJ44GICmpramtrAnjgYOK2tramJ44GCPmramps"+
6337 "Cm044GICmtq2prbnAnjgOk2prazwJ44GICmpramtrAnjgYOK2tramJ44GCPmramps"+
6338 "Cm044GICmtq2prbnAnjgOk2prazwJ44GICmpramtrAnjgYOK2tramJ44GCPmramps"+
6339 "aztca2is1CeBqApmrtacBqppq2prbnAnjgOk2prazwJ44GICmpramtrAnjgYOK2tramJ44GCPmramps"+
6340 "banjg4Kamtra2vJ44GCMpr2pAnjgYKamtra2psCeO1Ap2tsceO1qamsmcCeO1qpega2pq2wJ44KCCmtqamwJ"+
6341 "44KCCmtq2pAnjgYKamtra2psCeO1Ap2tsceO1qamsmcCeO1qpega2pq2wJ44KCCmtqamwJ"+
6342 "amw44GICmtq2pAnjgYKamtra2psCeO1Ap2tsceO1qamsmcCeO1qpega2pq2wJ44KCCmtqamwJ"+
6343 "44KCCmtq2prbnAnjgYKamtra2psCeO1Ap2tsceO1qamsmcCeO1qpega2pq2wJ44KCCmtqamwJ"+
6344 "gIEK";
6345 //</span>
6346
6347 //</span>
6348
6349 <style id="gsh-references-style">
6350 #references details { font-family:Georgia; }
6351 #references a { font-family:Georgia; }
6352 .wrap { white-space:normal; }
6354 <details id="references"><summary>References</summary><div class="gsh-src">
6355 Web technology
6357 <a href="https://html.spec.whatwg.org">HTML: The Living Standard</a> (September 2020)
6358 <a href="https://html.spec.whatwg.org/dev">Developer Version</a>
6359
6360 <a href="https://drafts.csswg.org">CSS Working Group Editor Drafts</a> (September 2020)
6361
6362 <a href="https://developer.mozilla.org/ja/docs/Web">MDN: web docs</a>
6363 <a href="https://developer.mozilla.org/ja/docs/Web/HTML/Element">HTML</a>
6364 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Selectors">CSS Selectors</a>
6365 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/Background-Repeat">repeat</a>
6366 <a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript/JavaScript_in_Everything">JavaScript</a>
6367 <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP/HTTP">HTTP</a>
6368 <a href="https://mdn-web-dna.s3-us-west-2.amazonaws.com/MDN-Browser-Compatibility_Report_2020.pdf">MDN Browser Compatibility Report (2020)</a>
6370
6371 Go language (August 2020 / Go 1.15)
6372 <a href="https://golang.org/doc/effective_go#Programming_Language">Effective Go Programming Language</a>
6373 <a href="https://golang.org/pkg/>Packages</a>
6374 <a href="https://godoc.org/x/net/websocket">WebSocket</a>
6375
6376 <a href="https://stackoverflow.com">Stackoverflow</a>
6377 <!--
6378 <iframe src="https://golang.org" width="100%" height="300"></iframe>
6379 </div></details>
6380 /*
6381 <details id="html-src" onclick="frame_open();"><summary>Raw Source</summary><div>
6382 <!-- h2>the full of this HTML including the Go code is here.</h2 -->
6383 <details id="gsh-whole-view"><summary>Whole file</summary>
6384 <a name="whole-src-view"></a>
6385 <span id="src-frame"></span><!-- a window to show source code -->
6386 </details>
6387 <details id="gsh-style-frame" onclick="fill_CSSView()"><summary>CSS part</summary>
6388 <a name="style-src-view"></a>
6389 <span id="gsh-style-view"></span>
6390 </details>
6391 <details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
6392 <a name="script-src-view"></a>
6393 <span id="gsh-script-view"></span>
6394 </details>
6395 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Built-in data part</summary>
6396 <a name="gsh-data-frame"></a>
6397 <span id="gsh-data-view"></span>
6398 </details>
6399 <div></div><details>
6400 <!--
6401 <div id="GshFooter0"></div>
6411 <!-- 2020-09-17 SatorixS, visible script ( -- >
6412 <div id="gjs_outerHTML" style="position:script;summary">
6413 <style>.gjscript { font-family:Georgia; }</style>
6414 <pre id="gjscript_1" class="gjscript">function gitest1(){ alert('Hello GJSscript!'); }
6415 <gitest1()
6416 </pre>
6417 <script>
6418 <!-- document.getElementById('gjscript_1');
6419 //eval(gjs.innerHTML,'
6420 //gjs_outerHTML = '';
6421 </script>
6422 <details><!-- ----- END-OF-VISIBLE-PART ----- >
6423 </details>
6424 <!--
6425 // 2020-09-06 added,
6426 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
6427 https://developer.mozilla.org/en-US/docs/Web/CSS/position
6428 -->
6429 <span id="GshGrid">(_)/<small>(Hit j k l h)</small></span>
6430 <span id="GStat"><br>
6431 <span id="GMenu" onclick="GShellMenu(this)" draggable="true"></span>
6432 <span id="GTop"></span>
6433 <div id="GShellPlane" onclick="showGShellPlane();"></div>
6434 <div id="RawTextViewer"></div>
6435 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
6436 <style id="GshStyleDef">
6440 #line-numbered table,tr,td {
6441 margin:0;
6442 padding:4px;
6443 spacing:0;
6444 border:1px;
6445 }
6446 textarea.LineNumber {
6447 font-size:12px;
6448 font-family:monospace,Courier New;
6449 color:#222;
6450 padding:4px;
6451 text-align:right;
6452 }
6453 textarea.LineNumbered {
6454 font-size:12px;
6455 font-family:monospace,Courier New;
6456 padding:4px;
6457 wrap:off;
6458 }
6459 #RawTextViewer{
6460 z-index:0;
6461 position:fixed; top:0px; left:0px;
6462 width:100%; xxheight:50px; xheight:0px;
6463 overflow:auto;
6464 color:#fff; background-color:rgba(128,128,256,0.2);
6465 font-size:10px; font-family:Georgia;
6466 spellcheck:false;
6467 }
6468 #RawTextViewerClose{
6469 z-index:0;
6470 position:fixed; top:-100px; left:-100px;
6471 color:#fff; background-color:rgba(128,128,256,0.2);
6472 font-size:10px; font-family:Georgia;
6473 white-space:pre;
6474 }
6475 #xxGShellPlane{
6476 z-index:0;
6477 position:fixed; top:0px; left:0px;
6478 width:100%; height:0px;
6479 overflow:auto;

```



```

7128     color:#ffff; background-color:rgba(0,64,0.7);
7129     text-align:left;
7130     vertical-align:middle;
7131   }
7132 </style>
7133
7134 <script id="gsh-script">
7135 // 2013_0909 added, permanet local storage
7136 // https://Developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
7137 var MyHistory = null;
7138 Permanent.localstorage;
7139 MyHistory = Permanent.getItem('MyHistory');
7140 if( MyHistory == null ){ MyHistory = "" }
7141 d = new Date();
7142 MyHistory += gettime() / 1000 + " " + document.URL + "\n" + MyHistory
7143 Permanent.setItem('MyHistory', MyHistory)
7144 //Permanent.setitem('MyHistory', MyHistory)
7145
7146 var GJLog_Win = null
7147 var GJLog_Tab = null
7148 var GJLog_Stat = null
7149 var GJLog_Text = null
7150 var GJWin_Mode = null
7151 var FProductInterval = 0
7152
7153 var GJ_FactoryID = -1
7154 var GJFactory = null
7155 if( e = document.getElementById('GJFactory_0') ){
7156   GJfactory_1.height = 0
7157   GJfactory = e
7158   e.setAttribute('class','GJFactory')
7159   var GJ_FACTORYID = 0
7160 }else{
7161   GJfactory = GJfactory_1
7162   var GJ_FACTORYID = 1
7163 }
7164
7165 function GJFactory_Destroy(){
7166   gif = GJfactory
7167   //gif = document.getElementById('GJFactory')
7168   //alert('gif'+=+gif)
7169   if( gif != null ){
7170     if( gif.childNodes != null ){
7171       for( i = 0; i < gif.childNodes.length; i++ ){
7172         gif.removeChild(gif.childNodes[i])
7173       }
7174     }
7175     gif.innerHTML = ''
7176     gif.style.width = 0
7177     gif.style.height = 0
7178     gif.removeAttribute('style')
7179     GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
7180     window.clearInterval(FProductInterval)
7181     return '-- Destroy: work product destroyed'
7182   }else{
7183     return '-- Destroy: work product not exist'
7184   }
7185 }
7186
7187 var TransMode = false
7188 var OnkeyControl = false
7189 var OnKeyShift = false
7190 var OnKeyAlt = false
7191 var OnKeyF = false
7192 var OnKeyK = false
7193 var OnKeyL = false
7194
7195 function GJWin_OnkeyUp(ev){
7196   keycode = ev.code;
7197   if( keycode == 'ShiftLeft' ){
7198     OnKeyShift = false
7199   }else{
7200     if( keycode == 'ControlLeft' ){
7201       OnKeyControl = false
7202     }else{
7203       if( keycode == 'AltLeft' ){
7204         OnKeyAlt = false
7205       }else{
7206         if( keycode == 'KeyJ' )( OnKeyJ = false )else
7207           if( keycode == 'KeyK' )( OnKeyK = false )else
7208             if( keycode == 'KeyL' )( OnKeyL = false )else
7209           {
7210             ev.preventDefault()
7211           }
7212         function and(a,b){ if(a){ if(b){ return true; } return false; } }
7213         function GJWin_OnkeyDown(ev){
7214           keycode = ev.code;
7215           mode = '';
7216           key = '';
7217           if( keycode == 'ControlLeft' ){
7218             OnKeyControl = true
7219             ev.preventDefault()
7220             return;
7221           }else{
7222             if( keycode == 'ShiftLeft' ){
7223               OnKeyShift = true
7224               ev.preventDefault()
7225               return;
7226             }else{
7227               if( keycode == 'AltLeft' ){
7228                 ev.preventDefault()
7229                 OnKeyAlt = true
7230                 return;
7231               }else{
7232                 if( keycode == 'Backquote' ){
7233                   TransMode = !TransMode
7234                   ev.preventDefault()
7235                 }else{
7236                   if( and(keycode == 'Space', OnKeyShift) ){
7237                     TransMode = !TransMode
7238                     ev.preventDefault()
7239                   }else{
7240                     if( keycode == 'ShiftRight' ){
7241                       TransMode = !TransMode
7242                     }else{
7243                       if( keycode == 'Escape' ){
7244                         TransMode = true
7245                         ev.preventDefault()
7246                       }else{
7247                         if( keycode == 'Enter' ){
7248                           TransMode = false
7249                           //ev.preventDefault()
7250                         }else{
7251                           if( ev.altKey )( key += 'Alt+' )
7252                           if( onKeyControl )( key += 'Ctrl+' )
7253                           if( OnKeyShift )( key += 'Shift+' )
7254                           if( and(keycode != 'KeyJ', OnKeyJ) )( key += 'J+' )
7255                           if( and(keycode != 'KeyK', OnKeyK) )( key += 'K+' )
7256                           if( and(keycode != 'KeyL', OnKeyL) )( key += 'L+' )
7257                           key += keycode
7258                         }
7259                         if( TransMode ){
7260                           //mode = "[\u0343\u2013\u0202]";
7261                           JAAutf8 = String.fromCharCode([0343,0201,0202]);
7262                           utf8dec = new TextDecoder('utf-8');
7263                           JAa = utf8dec.decode(JAAutf8);
7264                           mode = "[" + JAa + "]";
7265                         }
7266                       }
7267                     }
7268                   }
7269                 }
7270               }
7271             }
7272           }
7273         }
7274       }
7275     }
7276   }
7277   //---- /gjmode.innerHTML = "[---]"
7278   GJWin_Mode.innerHTML = mode + '' + key
7279   //alert('key'+keycode)
7280   ev.stopPropagation()
7281   //ev.preventDefault()
7282 }
7283 function GJWin_OnScroll(ev){
7284   x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
7285   y = DragStatty = gsh.getBoundingClientRect().top.toFixed(0)
7286   GJLog_append('OnScroll: x=' + x + ',y=' + y)
7287 }
7288 document.addEventListener('scroll',GJWin_OnScroll)
7289 function GJWin_ONResize(ev){
7290   w = window.innerWidth

```

```

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

```

```

7452 gjtext = GJLog_Text = document.createElement('textareा')
7453 gjtext.addEventListener('keydown',GJWin_OnKeydown)
7454 gjtext.addEventListener('keyup',GJWin_OnKeyUp)
7455 gjtext.addEventListener('resize',GJWin_OnResizeTextarea)
7456 gjtext.id = wid + '_Text'
7457 gjtext.setAttribute('class','GJText')
7458 gjtext.setAttribute('spellcheck','false')
7459 GJWin.appendChild(gjtext)
7460
7461
7462
7463 // user's mode as of IME
7464 gjmode = GJWin_Mode = document.createElement('textareा')
7465 gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7466 gjmode.addEventListener('keyup',GJWin_OnKeyUp)
7467 gjmode.id = wid + '_Mode'
7468 gjmode.setAttribute('class','GJMode')
7469 gjmode.setAttribute('spellcheck','false')
7470 gjmode.innerHTML = '---'
7471 GJWin.appendChild(gjmode)
7472
7473 gjwin.zIndex = 30000
7474 GJFactory.appendChild(gjwin)
7475
7476 gjtab.scrollTop = 0
7477 gjstat.scrollTop = 0
7478
7479 //x = gwin.getBoundingClientRect().left.toFixed(0)
7480 //y = gwin.getBoundingClientRect().top.toFixed(0)
7481 //gwin.style.position = 'static'
7482 //gwin.style.left = 0
7483 //gwin.style.top = 0
7484
7485 //update = '({wid}.value=ateShort())';
7486 update = '(GJ_showTime1('+wid+'))';
7487 //2020-09-19 this causes memory leaks
7488 //FProductInterval = window.setInterval(update,200)
7489 //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
7490 //FProductInterval = window.setInterval(GJ_showTime1,200,wid);
7491 FProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
7492 return update
7493 }
function xxxCF_Stat1cClass(){
7494 GJLog_Min.style.removeProperty('width')
7495 GJLog_Tab.style.removeProperty('width')
7496 GJLog_Stat.style.removeProperty('width')
7497 GJLog_Text.style.removeProperty('width')
7498 return 'Stripped classes'
7499 }
function islem(id){
7500 return document.getElementById(id) != null
7501 }
function GJLog_append(...args){
7502 txt = GJLog_Text;
7503 if (txt == null){
7504     return; // maybe GJLog element is removed
7505 }
7506 logs = args.join(' ');
7507 txt.value += logs + '\n';
7508 txt.scrollTop = txt.scrollHeight
7509 //GJLog_Status = DateShort()
7510 //window.addEventListener('time',GJLog_StatUpdate)
7511 function test_GJ_Console(){
7512     window.setInterval(GJLog_StatUpdate,1000);
7513     GJConsole = 'GJ_Console'
7514     e = GJFactory();
7515     console.log('GJFO #' + e.id' from w=' + e.style.width + ', h=' + e.style.height)
7516     e.style.width = 360; //GJSize
7517     e.style.height = 320;
7518     console.log('GJFO #' + e.id' to w=' + e.style.width + ', h=' + e.style.height)
7519 }
7520 //// test_GJ_Console();
7521
7522 var StopConsoleLog = true
7523 // 2020-09-15 added,
7524 // log should be saved to permanent memory
7525 //console = new Proxy(console.log,{ alert() })
7526 _console_log = console.log
7527 _console_info = console.info
7528 _console_warn = console.warn
7529 _console_error = console.error
7530 _console_exception = console.exception
7531 // above functions are inlined
7532 console.exception = function(...args){
7533     _console_exception(...args)
7534     alert('-- got console.exception("'+args+'")')
7535 }
7536 console.error = function(...args){
7537     _console_error(...args)
7538     alert('-- got console.error("'+args+'")')
7539 }
7540 console.warn = function(...args){
7541     _console_warn(...args)
7542     alert('-- got console.warn("'+args+'")')
7543 }
7544 console.info = function(...args){
7545     _console_info(...args)
7546     alert('-- got console.info("'+args+'")')
7547 }
7548 _console_info = function(...args){
7549     alert('-- got console.info("'+args+'")')
7550 }
7551 _console_info(...args)
7552 console.xxlog = function(...args){ // rewrite xxlog to log to enable it
7553     _console_log(...args)
7554     if(StopConsoleLog){
7555         return;
7556     }
7557     if( 0 <= args[0].indexOf('!') ){
7558         //alert('-- got console.log("'+args+'")')
7559     }
7560 }
7561 GJLog_append(...args)
7562
7563 //document.getElementById('GshFaviconURl').href = GShellFavicon
7564 //document.getElementById('GshFaviconURl').href = GShellInsideIcon
7565 //document.getElementById('GshFaviconURl').href = ITSMoreQR
7566 //document.getElementById('GshFaviconURl').href = GSellLogo
7567
7568 // id of GShell HTML elements
7569 var E_BANNER = "GshBanner" // banner element in HTML
7570 var E_FOOTER = "GshFooter" // footer element in HTML
7571 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7572 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7573 var E_TODO = "gsh-todo" // TODO of GShell
7574 var E_DICT = "gsh-dict" // Dictionary of GShell
7575
7576 function bannerElem(){ return document.getElementById(E_BANNER); }
7577 function bannerStyleFunc(){ return bannerElem().style; }
7578 var bannerStyle = bannerStyleFunc()
7579 function GshSetImages(){
7580     document.getElementById('GshFaviconURl').href = GShellInsideIcon
7581     bannerStyle.backgroundImage = "url("+GShellLogo+"")";
7582     //bannerStyle.backgroundImage = "url("+GShellInsideIcon+"")";
7583     //bannerStyle.backgroundImage = "url("+GShellFavicon+"")";
7584     //Menu.style.backgroundImage = "url("+GShellInsideIcon+"")";
7585     //showFooter();
7586 }
7587 function GshInsideIconSetup(){
7588     GMenu.style.backgroundImage = "url("+GShellInsideIcon+"")";
7589     GMenu.style.zIndex = 1000000;
7590     //GMenu.style.left = window.innerWidth - 10
7591     GMenu.style.left = 0;
7592     GMenu.style.top = window.innerHeight - 90; // - 200
7593     window.addEventListener('resize',GshInsideIconSetup);
7594 }
7595
7596 function footerElem(){ return document.getElementById(E_FOOTER); }
7597 function footerStyle(){ return footerElem().style; }
7598 //footerElem().style.backgroundImage="url("+ITSMoreQR+"")";
7599 //footerStyle().backgroundImage = "url("+ITSMoreQR+"")";
7600
7601 function html_fold(e){
7602     if( e.innerHTML == "Fold" ){
7603         e.innerHTML = "Unfold"
7604         document.getElementById('gsh-menu-exit').innerHTML=""
7605         GshFeatures.open = false
7606         document.getElementById('GshStatement').open=false
7607         document.getElementById('html-src').open=false
7608         document.getElementById('GINDEX').open=false
7609         document.getElementById('GOCODE').open=false
7610         document.getElementById('TODO').open=false
7611         document.getElementById('references').open=false
7612     }else{
7613         e.innerHTML = "Fold"
7614     }
7615 }

```

```

7614     document.getElementById('GshStatement').open=true
7615     GshFeatureA.style.display = true
7616     document.getElementById(E_GINDEX).open=true
7617     document.getElementById(E_GCODE).open=true
7618     document.getElementById(E_TODO).open=true
7619     document.getElementById('references').open=true
7620   }
7621   function html_pure(e){
7622     if( e.innerHTML == "Pure" ){
7623       document.getElementById('gsh').style.display=true
7624       //document.style.display = false
7625       e.innerHTML = "Unpure"
7626     }else{
7627       document.getElementById('gsh').style.display=false
7628       //document.style.display = true
7629       e.innerHTML = "Pure"
7630     }
7631   }
7632 }
7633
7634 var bannerIsStopping = false
7635 //NOTE: .com/ISREF/prop_style_backgroundposition.asp
7636 function shiftBG(){
7637   bannerIsStopping = !bannerIsStopping
7638   bannerStyle.backgroundPosition = "0 0";
7639 }
7640 // status should be inherited on Window Fork(), so use the status in DOM
7641 function html_stop(e,toggle){
7642   if( toggle ){
7643     if( e.innerHTML == "Stop" ){
7644       bannerIsStopping = true
7645       e.innerHTML = "Start"
7646     }else{
7647       bannerIsStopping = false
7648       e.innerHTML = "Stop"
7649     }
7650   }else{
7651     // update JavaScript variable from DOM status
7652     if( e.innerHTML == "Stop" ){ // shown if it's running
7653       bannerIsStopping = false
7654     }else{
7655       bannerIsStopping = true
7656     }
7657   }
7658 }
7659 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
7660 //html_stop(bannerElem),false) // onInit.
7661
7662 https://www.w3schools.com/jsref/met_win_setinterval.asp
7663 var banNshift = 0;
7664 function conslog(str){
7665   //console.log(str);
7666 }
7667 function shiftBanner(){
7668   var now10 = new Date().getTime();
7669   bpos = ((now10/1000).toFixed(0))+'px' + " 0px";
7670   if( ibannerIsStopping ){
7671     bannerStyle.backgroundPosition = bpos;
7672     //GshBanner.style.setProperty('background-position',bpos,'important');
7673     banNshift += 1;
7674     conslog('shiftBanner <'+GshBanner.nodeName+'> '+banNshift
7675     + ' pos='+(now10)
7676     //+ ' stop'+bannerIsStopping
7677     + ' pos='+bpos
7678     + ' -> '+bannerStyle.backgroundPosition);
7679   }
7680 }
7681 function Banner_init(){
7682   console.log('-- Banner Shift init.');
7683   window.setInterval(shiftBanner,10); // onInit.
7684 }
7685
7686 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
7687 // from embedded html to standalone page
7688 var Mychildren = 0
7689 function html_fork(){
7690   ResetterPerf();
7691   Resetter();
7692   Reset_ShadingCanvas();
7693   GJfactory_Destroy();
7694   Mychildren += 1
7695   WinId = document.getElementById('gsh-WinId').innerHTML + "." + Mychildren;
7696   newwin = window.open("",WinId,"");
7697   newwin.document.getElementById('gsh');
7698   srchtml = src.outerHTML;
7699   newwin.document.write("<*<html>\n");
7700   newwin.document.write(srchtml);
7701   newwin.document.write("<*<html>\n");
7702   newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
7703   newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
7704   newwin.document.close();
7705   newwin.focus();
7706 }
7707 function html_close(){
7708   window.close()
7709 }
7710 function win_jump(win){
7711   //win = window.top;
7712   win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
7713   if( win == null ){
7714     console.log('jump to window.opener("'+win+'") (Error)\n')
7715   }else{
7716     console.log('jump to window.opener("'+win+'")\n')
7717     win.focus();
7718   }
7719 }
7720
7721 // 0.2.9 2020-0902 created checksum of HTML
7722 CRC32UNIX = 0x04C11DB7 // Unix cksum
7723 function byteCRC32add(bigcrc,octstr,octlen){
7724   var crc = new Int32Array(1)
7725   crc[0] = bigcrc
7726
7727   let oit = 0
7728   for( ; oit < octlen; oit++ ){
7729     var oct = new Int8Array(1)
7730     oct[0] = octstr[oit]
7731     for( let bi=0; bi<8; bi++ ){
7732       //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oit+"."+bi+"]\n")
7733       ovf1 = crc[0] < 0 ? 1 : 0
7734       ovf2 = oct[0] < 0 ? 1 : 0
7735       ovf = ovf1 ^ ovf2
7736       oct[0] <<= 1
7737       crc[0] <<= 1
7738       if( ovf ){ crc[0] ^= CRC32UNIX }
7739     }
7740   }
7741   //console.log("--CRC32 byteAdd return crc="+crc[0]+","+oit+"/"+octlen+"\n")
7742   return crc[0];
7743 }
7744 function strCRC32add(bigcrc,str1,strlen){
7745   var crc = new Uint32Array(1)
7746   var code = new Uint8Array(strlen);
7747   for( i = 0; i < strlen; i++){
7748     code[i] = str1.charCodeAt(i) // not charAt() !!!!!
7749     //console.log("== "+code[i].toString(16)+" <== "+str1[i]+"\n")
7750   }
7751   crc[0] = byteCRC32add(crc,code,strlen)
7752   //console.log("--CRC32 strAdd return crc="+crc[0]+\n")
7753   return crc[0];
7754 }
7755 function byteCRC32end(bigcrc,len){
7756   var crc = new Uint32Array(1)
7757   crc[0] = bigcrc
7758   var slen = new Uint32Array(4)
7759   let li = 0
7760   for( ; li < 4; ){
7761     slen[li] = len
7762     li += 1
7763     len >>= 8
7764     if( len == 0 ){
7765       break
7766     }
7767   }
7768   crc[0] = byteCRC32add(crc[0],slen,li)
7769   crc[0] ^= 0xFFFFFFFF
7770   return crc[0];
7771 }
7772 function strCRC32(str1,len){
7773   var crc = new Uint32Array(1)
7774   crc[0] = 0
7775 }
```

```

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

```

```

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

```

```

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

```

```

8262     return '0' + n;
8263   }else{
8264     return n;
8265   }
8266 }
8267 function DateHourMin(){
8268   d = new Date();
8269   //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes());
8270   return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
8271 }
8272 function DateShort0(d){
8273   return d.getFullYear() +
8274     '/' + zero2(d.getMonth())
8275     + '/' + zero2(d.getDate())
8276     + ':' + zero2(d.getHours())
8277     + ':' + zero2(d.getMinutes())
8278     + ':' + zero2(d.getSeconds())
8279 }
8280 function DateShort(){
8281   return DateShort0(new Date());
8282 }
8283 function DateLong0(ms){
8284   d = new Date();
8285   d.setTime(ms);
8286   return DateShort0(d) +
8287     '.' + d.getMilliseconds() +
8288     '.' + d.getTimezoneOffset()/60 +
8289     '.' + d.getTime() + '.' + d.getMilliseconds()
8290 }
8291 function DateLong(){
8292   return DateLong0(new Date());
8293 }
8294 function GShellMenu(e){
8295   //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
8296   //showGShellPlane()
8297   ConfigClick();
8298 }
8299 // placements of planes
8300 function GShellResize(ev){
8301   //if( document.getElementById("GMenu") != null ){
8302   //  GshInsideIconSetup();
8303   //  GMENU.style.left = window.innerWidth - 100
8304   //  GMENU.style.top = window.innerHeight - 90 - 200
8305   //  //console.log('place GMENU '+GMENU.style.left+' '+GMENU.style.top)
8306   //}
8307   GStat.style.width = window.innerWidth
8308   //if( document.getElementById("GPos") != null ){
8309   //  GPos.style.width = window.innerWidth
8310   //  GPos.style.top = window.innerHeight - 30; //GPos.style.height
8311   //}
8312   //if( document.getElementById("GLog") != null ){
8313   //  GLog.style.width = window.innerWidth
8314   //  GLog.innerHTML = ""
8315   //}
8316   //if( document.getElementById("GLog") != null ){
8317   //  GLog.innerHTML = "Resize: w=" + window.innerWidth +
8318   //  ", h=" + window.innerHeight
8319   //}
8320   //showCurElementPosition(ev)
8321 }
8322 function GShellResizeX(){
8323   GShellResizeX("[RESIZE]")
8324 }
8325 window.onresize = GShellResize
8326 var prevNode = null;
8327 var LogMouseMoveOverElement = false;
8328 function GJSH_OnMouseMove(ev){
8329   if( LogMouseMoveOverElement == false ){
8330     return;
8331   }
8332   x = ev.clientX
8333   y = ev.clientY
8334   d = new Date()
8335   t = d.getTime() / 1000
8336   if( document.elementFromPoint ){
8337     e = document.elementFromPoint(x,y)
8338     if( e != null ){
8339       if( e == prevNode ){
8340         console.log('Mo-' + t + '(' + x + ',' + y + ') ' +
8341           'e.nodeType=' + e.tagName + '#' + e.id)
8342         prevNode = e
8343       }
8344     }else{
8345       console.log(t + '(' + x + ',' + y + ') no element')
8346     }
8347   }else{
8348     console.log(t + '(' + x + ',' + y + ') no elementFromPoint')
8349   }
8350 }
8351 window.addEventListener('mousemove',GJSH_OnMouseMove);
8352
8353 function GJSH_OnMouseMoveScreen(ev){
8354   x = ev.clientX
8355   y = ev.clientY
8356   d = new Date()
8357   t = d.getTime() / 1000
8358   console.log(t + '(' + x + ',' + y + ') no elementFromPoint')
8359 }
8360 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
8361
8362 function ScrollToElement(oe,ne){
8363   ne.scrollIntoView()
8364   ny = ne.getBoundingClientRect().top.toFixed(0)
8365   nx = ne.getBoundingClientRect().left.toFixed(0)
8366   //GLog.innerHTML = "["+ny+","+nx+"]"
8367   //window.scrollTo(nx,ny)
8368
8369   GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
8370   GshGrid.style.left = "250px";
8371   GshGrid.style.zIndex = 0
8372   if( fail == true ){
8373     oy = oe.getBoundingClientRect().top.toFixed(0)
8374     ox = oe.getBoundingClientRect().left.toFixed(0)
8375     y = e.getBoundingClientRect().top.toFixed(0)
8376     x = e.getBoundingClientRect().left.toFixed(0)
8377     window.scrollTo(ox,y)
8378     ny = oe.getBoundingClientRect().top.toFixed(0)
8379     nx = e.getBoundingClientRect().left.toFixed(0)
8380     //GLog.innerHTML = "["+oy+","+ox+"]->["+y+","+x+"]->["+ny+","+nx+"]"
8381   }
8382 }
8383
8384 function showGShellPlane(){
8385   if( GShellPlane.style.zIndex == 0 ){
8386     GShellPlane.style.zIndex = 1000;
8387     GShellPlane.style.left = 30;
8388     GShellPlane.style.height = 320;
8389     GShellPlane.innerHTML = DateLong() + "<br>" +
8390     "- History --> or " + MyHistory;
8391   }else{
8392     GShellPlane.style.zIndex = 0;
8393     GShellPlane.style.left = 0;
8394     GShellPlane.style.height = 50;
8395     GShellPlane.innerHTML = "";
8396   }
8397 }
8398
8399 var SuppressGJShell = false
8400 function jgshCommand(keyevent){
8401   if( SuppressGJShell ){
8402     return
8403   }
8404   key = keyevent
8405   keyCode = key.keyCode
8406   //GStat.style.width = window.innerWidth
8407   GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
8408
8409   console.log("JGSh-KeyPress:" +keycode+"(^~)//")
8410   if( keyCode == "Slash" ){
8411     console.log('(' + x + ',' + y + ') ')
8412     e = document.elementFromPoint(x,y)
8413     console.log(' (' + x + ',' + y + ') ' + e.nodeType + ' ' + e.tagName + '#' + e.id)
8414   }
8415   if( keyCode == "Digit0" ){ // fold side-bar
8416     // "zero page"
8417     showGShellPlane();
8418   }
8419   if( keyCode == "Digit1" ){ // fold side-bar
8420     primary.style.width = "44%"
8421     secondary.style.width = "0%"
8422     secondary.style.opacity = 0
8423     GStat.innerHTML = "[Single Column View]"

```

```

8424     else
8425       if( keycode == "Digit2" ){ // unfold side-bar
8426         primary.style.width = "38"
8427         secondary.style.width = "38"
8428         secondary.style.opacity = 1
8429         GStat.innerHTML = "[Double Column view]"
8430       }
8431     if( keycode == "KeyU" ){ // fold/unfold all
8432       html_fold(GshMenuFold);
8433       location.href = "#"+CurElement.id;
8434     }
8435     if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8436       Curlement.open = !CurElement.open;
8437     }
8438     if( keycode == "ArrowLeft" ){ // unfold the element
8439       Curlement.open = true
8440     }
8441     if( keycode == "ArrowLeft" ){ // unfold the element
8442       Curlement.open = false
8443     }
8444     if( keycode == "KeyI" ){ // inspect the element
8445       e = CurElement
8446       //GLog.innerHTML =
8447       GJLog.append("Current Element: " + e + "<br>" +
8448         + "name=" + e.nodeName + ", " +
8449         + "id=" + e.id + ", " +
8450         + "children=" + e.childNodes.length + ", " +
8451         + "parent=" + e.parentNode.id + "<br>" +
8452         + "text=" + e.textContent)
8453       GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8454       return
8455     }
8456     if( keycode == "KeyM" ){ // memory the position
8457       MemElement = CurElement
8458     }
8459     if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8460       e = nextElement(Curlement)
8461       if( e != null ){
8462         setColor(e,"SUMMARY","#fff")
8463       oe = Curlement
8464       Curlement = e
8465       //location.href = "#"+e.id;
8466       ScrollToElement(oe,e)
8467     }
8468   }
8469   if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8470     oe = Curlement
8471     e = previousCurElement()
8472     if( e != null ){
8473       setColor(e,"SUMMARY","#fff")
8474       setColor(e,"SUMMARY","#f8f") // should be complement
8475     oe = Curlement
8476     Curlement = e
8477     //location.href = "#"+e.id;
8478     ScrollToElement(oe,e)
8479   }
8480   else
8481     e = document.getElementById("GshBanner")
8482     if( e != null ){
8483       setColor(CurElement,"SUMMARY","#fff")
8484       CurElement = e
8485       ScrollToElement(oe,e)
8486     }
8487     else
8488       e = document.getElementById("primary")
8489       if( e != null ){
8490         setColor(CurElement,"SUMMARY","#fff")
8491       CurElement = e
8492       ScrollToElement(oe,e)
8493     }
8494   }
8495   if( keycode == "KeyR" ){
8496     location.reload()
8497   }
8498   if( keycode == "KeyJ" ){
8499     GshGrid.style.top = '120px';
8500     GshGrid.innerHTML = '<>{Down}';
8501   }
8502   if( keycode == "KeyK" ){
8503     GshGrid.style.top = '0px';
8504     GshGrid.innerHTML = '{--}{Up}';
8505   }
8506   if( keycode == "KeyH" ){
8507     GshGrid.style.left = '0px';
8508     GshGrid.innerHTML = '{_}{Left}';
8509   }
8510   if( keycode == "KeyL" ){
8511     //GLog.innerHTML +
8512     GJLog.append(
8513       'screen=' + screen.width + 'px' + '<br>' +
8514       'window=' + window.innerWidth + 'px' + '<br>' +
8515     )
8516   GshGrid.style.left = (document.documentElement.clientWidth - 160).toString(10) + 'px';
8517   GshGrid.innerHTML = '(@_){Right}';
8518   }
8519   if( keycode == "KeyS" ){
8520     html_stop(GshMenuStop,true)
8521   }
8522   if( keycode == "KeyF" ){
8523     html_fork()
8524   }
8525   if( keycode == "KeyC" ){
8526     window.close()
8527   }
8528   if( keycode == "KeyD" ){
8529     html_digest()
8530   }
8531   if( keycode == "KeyV" ){
8532     e = document.getElementById('gsh-digest')
8533     if( e != null ){
8534       showDigest(e)
8535     }
8536   }
8537   showCurElementPosition("[+key.code+]" --);
8538   //if( document.getElementById("GPos") != null ){
8539   //  GPos.innerHTML += "[+key.code+]" --
8540   //}
8541   //GShellResizeX("[+key.code+] --");
8542 }
8543 var initGSKC = false;
8544 function GShell_initKeyCommands(){
8545   if( initGSKC ) return; initGSKC = true;
8546   GShellResizeX("[INT]");
8547   DisplaySize = '-- Display' +
8548     + 'screen=' + screen.width + 'px, +' + 'window=' + window.innerWidth + 'px';
8549   GLog.innerHTML += GShell_resize();
8550   GJLog.append(
8551     '-- Gshell: ' + GshVersion.innerHTML + '\n' +
8552     '-- Digest: ' + digest + '\n' +
8553     DisplaySize +
8554     //+ "cbr" + "-- LastVisit:<br>" + MyHistory
8555   );
8556   GShellResizeX(null);
8557 }
8558 //GShell_initKeyCommands();
8559
860 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
861 //Convert a string into an ArrayBuffer
862 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
863 function str2ab(str) {
864   const buf = new ArrayBuffer(str.length);
865   const bufView = new Uint8Array(buf);
866   for (let i = 0, strLen = str.length; i < strLen; i++) {
867     bufView[i] = str.charCodeAt(i);
868   }
869   return buf;
870 }
871
872 function importPrivateKey(pem) {
873   const binaryDerString = window.atob(pemContents);
874   const binaryDerBuf = str2ab(binaryDerString);
875   return window.crypto.subtle.importKey(
876     "pkcs8",
877     binaryDer,
878     {
879       name: "RSA-PSS",
880       modulusLength: 2048,
881       publicExponent: new Uint8Array([1, 0, 1]),
882       hash: "SHA-256",
883     }
884   );
885 }

```

```

8586     },
8587     true,
8588     ["sign"]
8589   );
8590 }
8591 //importPrivateKey(ppem)
8592 //Key = {}
8593 //buf = "abc"
8594 //enc = "xxxxxxxx"; //crypto.publicEncrypt(key,buf)
8595 //b64 = btoa(enc)
8596 //dec = atob(b64)
8597 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8598 </script>
8599 </div>
8600
8601 //<!-- ===== Work { ===== -->
8602 <span id="PackmonGo_WorkCodeSpan">
8603   /*
8604    * Details id="PackmonGo_Section" open="">summary id="PackmonGo_Summary">PackmonGo</summary>
8605    <!--> PackmonGo // 2020-1025 SatoxITS { -->
8606    <h2>PackmonGo</h2>
8607    <div id="PackmonGo_1" class="PackmonGo">
8608      <canvas id="PackmonGo_1_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8609    </div>
8610    <div id="PackmonGo_2" class="PackmonGo">
8611      <canvas id="PackmonGo_2_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8612    </div>
8613    <div id="PackmonGo_3" class="PackmonGo">
8614      <canvas id="PackmonGo_3_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8615    </div>
8616    <div id="PackmonGo_4" class="PackmonGo">
8617      <canvas id="PackmonGo_4_Canvas" class="PackmonGo_Canvas" width="400px" height="400px" draggable="true"></canvas>
8618    </div>
8619  </style>
8620  <style>
8621 .PackmonGo {
8622   position:fixed !important;
8623   background-color:rgba(0,0,0,0.0);
8624 }
8625 .PackmonGo_Canvas {
8626   background-color:rgba(0,0,0,0.0);
8627 }
8628 </style>
8629 <script>
8630 var stopPackmonFlag = false;
8631 var PackmonInit = false;
8632 function spawnPackmonGo(){
8633   if( PackmonInit == false ){
8634     pgw = 100;
8635     pgh = 100;
8636     pgo = 0.5;
8637
8638     ctx = PackmonGo_1_Canvas.getContext('2d');
8639     ctx.fillStyle = 'rgba(255,255,0,'+pgo+')';
8640     ctx.fillRect(0,0,pgw,pgh);
8641     base = PackmonGo_1;
8642     gsh.appendChild(base);
8643     base.style.zIndex = 1000;
8644     base.style.position = "fixed";
8645     base.style.left = 0 +'px';
8646     base.style.top = 0 +'px';
8647     base.xinc = 4;
8648     base.yinc = 4;
8649     base.scrx = 0;
8650     base.scrx = 0;
8651     base.pgw = 100;
8652     base.pgh = 100;
8653     movePGwindow(PackmonGo_1);
8654
8655     ctx = PackmonGo_2_Canvas.getContext('2d');
8656     ctx.fillStyle = 'rgba(127,255,127,'+pgo+')';
8657     ctx.fillRect(0,0,pgw,pgh);
8658     base = PackmonGo_2;
8659     gsh.appendChild(base);
8660     base.style.zIndex = 1001;
8661     base.style.position = "fixed";
8662     base.style.left = 200 +'px';
8663     base.style.top = 0 +'px';
8664     base.xinc = 4;
8665     base.yinc = 4;
8666     base.scrx = 0;
8667     base.scrx = 0;
8668     base.pgw = 100;
8669     base.pgh = 100;
8670     movePGwindow(PackmonGo_2);
8671
8672     ctx = PackmonGo_3_Canvas.getContext('2d');
8673     pgo = 0.3;
8674     ctx.fillStyle = 'rgba(64,64,255,'+pgo+')';
8675     ctx.fillRect(0,0,pgw,pgh);
8676     base = PackmonGo_3;
8677     gsh.appendChild(base);
8678     base.style.zIndex = 1002;
8679     base.style.position = "fixed";
8680     base.style.left = 200 +'px';
8681     base.style.top = 0 +'px';
8682     base.xinc = 4;
8683     base.yinc = 4;
8684     base.scrx = 0;
8685     base.scrx = 0;
8686     base.pgw = 100;
8687     base.pgh = 100;
8688     movePGscreen(PackmonGo_3);
8689
8690     ctx = PackmonGo_4_Canvas.getContext('2d');
8691     pgw = pgh = 400;
8692     ctx.beginPath();
8693     ctx.strokeStyle = 'rgba(0,0,0,0)';
8694     ctx.arc(200,200,200,0,Math.PI*2,true);
8695     ctx.stroke();
8696     pgo = 0.4;
8697     ctx.fillStyle = 'rgba(255,31,32,'+pgo+')';
8698     ctx.fill();
8699     base = PackmonGo_4;
8700     gsh.appendChild(base);
8701     base.style.zIndex = 1002;
8702     base.style.position = "fixed";
8703     base.style.left = 200 +'px';
8704     base.style.top = 0 +'px';
8705     base.xinc = 4;
8706     base.yinc = 4;
8707     base.scrx = 0;
8708     base.scrx = 0;
8709     base.soff = 5000;
8710     base.pgw = pgw;
8711     base.pgh = pgh;
8712     movePGscreen(PackmonGo_4);
8713
8714   }function movePGwindow(base){
8715     if( stopPackmonFlag ){
8716       return;
8717     }
8718     x = parentInt(base.style.left);
8719     y = parentInt(base.style.top);
8720     w = window.innerWidth;
8721     h = window.innerHeight;
8722     if( x < 0 || w-base.pgw < x ){
8723       base.xinc = -base.xinc;
8724
8725       x += base.xinc;
8726       if( y < 0 || h-base.pgh < y ){
8727         base.yinc = -base.yinc;
8728         base.yinc = -base.yinc;
8729       }
8730       y += base.yinc;
8731       base.style.left = x +'px';
8732       base.style.top = y +'px';
8733       //console.log('PG x=' + x + ',y=' + y);
8734       //window.setTimeout(movePG,10);
8735     }
8736   }function movePGscreen(base){
8737     if( stopPackmonFlag ){
8738       return;
8739     }
8740     sw = screen.width;
8741     sh = screen.height;
8742     d = new Date();
8743     s = d.getTime(); // milli-seconds
8744     sx = ((s+base.soff) % 10000) * (screen.width / 10000);
8745     sy = ((s+base.soff) % 5000) * (screen.height / 5000);
8746     x = sx - window.screenX;
8747

```

```

8748     y = sy - window.screenY;
8749     base.style.left = x + 'px';
8750     base.style.top = y + 'px';
8751   }
8752   function movePcScreenCircle(base){
8753     if( stopPackmonFlag ){
8754       return;
8755     }
8756     sw = screen.width;
8757     sh = screen.height;
8758     pgw = base.offsetWidth;
8759     pgh = base.offsetHeight;
8760     d = new Date();
8761     s = d.getUTime(); // milli-seconds
8762     ds = s - basetime; // delay
8763     vsw = pgw + sw + pgw;
8764     vsh = pgh + sh + pgh;
8765     sx = -pgw + vsw * ((ds % 10000)/10000);
8766     sy = -pgi + vsh * ((ds % 10000)/10000);
8767     x = sx - window.screenX;
8768     y = sy - window.screenY;
8769     base.style.left = x + 'px';
8770     base.style.top = y + 'px';
8771   }
8772   function stopPackMon(){
8773     stopPackmonFlag = !stopPackmonFlag;
8774   }
8775   if( PackmonInit == false ){
8776     //window.setTimeout(movePG,mm300);
8777     window.setInterval(movePcGwindow,10,PackmonGo_1);
8778     window.setInterval(movePcGwindow,10,PackmonGo_2);
8779     window.setInterval(movePcGscreen,10,PackmonGo_3);
8780     window.setInterval(movePcScreen,10,PackmonGo_4);
8781     window.setInterval(movePcScreenCircle,10,PackmonGo_4);
8782     window.addEventListener('click',stopPackMon);
8783     PackmonInit = true;
8784   }
8785   function PackmonGo_Setup(){
8786     if( PackmonGo_Section.open == false ){
8787       spawnPackmonGo();
8788     }
8789   }
8790   PackmonGo_Summary.addEventListener('click',PackmonGo_Setup);
8791 if( PackmonGo_Section.open == true ){
8792   // spawnPackmonGo();
8793 }
8794 
```

```
</script>
```

```

8795 <input id="PackmonGo_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
8796 <input id="PackmonGo_WorkOpenCodeSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
8797 <input id="PackmonGo_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
8798 <span id="PackmonGo_WorkCodeScript"></span>
8799 <script id="PackmonGo_WorkCodeScript">
8800   function PackmonGo_OpenWorkCodeView(){
8801     PackmonGo_showWorkCode();
8802   }
8803   function PackmonGo_showWorkCode(){
8804     showHtmlCode(PackmonGo_WorkCodeView,PackmonGo_WorkCodeSpan);
8805   }
8806   PackmonGo_WorkCodeViewOpen.addEventListener('click',PackmonGo_showWorkCode);
8807   PackmonGo_OpenWorkCodeView(); // should be invoked by an event
8808 </script>
8809 <details>
8810   <!-- Template_WorkCodeSpan -->
8811   <!-- </span>
8812   <!-- ===== Work -->
8813   <!-- ===== Work -->
8814   <!-- ===== Work -->
8815   <!-- ===== Work -->
8816   <!-- ===== Work -->
8817   <!-- ===== Work { ===== -->
8818   <span id="SightGlass_WorkCodeSpan">
8819     <details id="SightClass"><summary>SightGlass</summary>
8820       <!-- ----- SightGlass // 2020-102 SatoxITS ( -->
8821       <h2>SightGlass</h2>
8822       <details><summary>Meter</summary>
8823         <div id="SightGlass_1_BPosition" class="SightGlass_TextData">(0000, 0000) (0000 x 0000) </div>
8824         <div id="SightGlass_1_BPositionOffset" class="SightGlass_TextData">(0000, 0000) </div>
8825         <div id="SightGlass_1_GPosition" class="SightGlass_TextData">(0000, 0000) (0000 x 0000) </div>
8826         <div id="SightGlass_1_BGPosition" class="SightGlass_TextData">(0000, 0000) (0000 x 0000) </div>
8827         <div id="SightGlass_1_Wheel" class="SightGlass_TextData">Wheel</div>
8828       </details>
8829       <p>
8830         <div id="SightGlass_1_Window" class="SightGlass_Window" draggable="true"></div>
8831       </p>
8832       <style>
8833         .SightGlass_TextData {
8834           color:#000;
8835           font-size:10pt;
8836         }
8837         .SightGlass_Window {
8838           xxzoom:2;
8839           resize:both;
8840           width:600px;
8841           height:220px;
8842           background-color:rgba(200,200,200,0.5);
8843           background-size:200px;
8844           xbackground-size:1280px 720px;
8845           background-image:url(WD-WallPaper03.png);
8846         }
8847       </style>
8848       <body>
8849         scroll-behavior:smooth;
8850       </body>
8851     </style>
8852     <script>
8853       // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
8854       var SGscrInity = 0;
8855       var SGscrInity = 0;
8856       var SG initx = 0;
8857       var SG inity = 0;
8858       var SG inity = 0;
8859       function SG_Adjust(){
8860         xy = 'xy(' + x + 'y' + ' ' + window.screenY;
8861         wh = window.innerWidth + ' ' + window.innerHeight;
8862         xywh = 'xy(' + xy + ') wh' + 'wh' + ']';
8863         if( SightGlass.open) SightGlass_1_BPosition.innerHTML = 'Browser: ' + xywh;
8864
8865         sg = SightGlass_1_Window;
8866         sgr = sg.getBoundingClientRect();
8867         xy = sgr.left.toFixed(0) + ' ' + sgr.top.toFixed(0);
8868         xy = sgr.width + 'x' + sgr.height;
8869         xywh = 'xy(' + xy + ') wh' + 'wh' + ']';
8870         if( SightGlass.open) SightGlass_1_BPosition.innerHTML = 'SightGlass: ' + xywh;
8871
8872         /SightGlass_1_Window.style.backgroundPosition = sgr.left+'px ' + sgr.top+'px';
8873         if( SG_initx == 0 ){
8874           SGscrInity = window.screenX;
8875           SGscrInity = window.screenY;
8876           //SightGlass_1_Window.style.backgroundSize = '200%';
8877           //SightGlass_1_Window.style.backgroundImage = 'url("WD-WallPaper03.png")';
8878           SG initx = sgr.left;
8879           SG inity = sgr.top;
8880         }
8881         dx = SG_initx - sgr.left;
8882         dy = SG_inity - sgr.top;
8883         dx = SG_initx - window.screenX;
8884         dy = SGscrInity - window.screenY;
8885         scrollf = 'Screen: '+sx+'dx '+sy+'dy';
8886         if( SightGlass.open) SightGlass_1_BGOffset.innerHTML = scrollf;
8887         dx += dx;
8888         dy += dy;
8889         bgpos = dx+'px ' + dy+'px';
8890         SightGlass_1_Window.style.backgroundPosition = bgpos;
8891         if( SightGlass.open) SightGlass_1_BGPosition.innerHTML = 'BGround: ' + SightGlass_1_Window.style.backgroundPosition;
8892       }
8893       var wheels = 0;
8894       function SG_wheel(e){
8895         wheels += 1;
8896         SightGlass_1_Wheel.innerHTML = 'Mouse Wheel: ' + wheels + ' #' + e.target.id;
8897         if( e.target.id == "SightGlass_1_Window" ){
8898           e.preventDefault();
8899         }
8900       }
8901     function SG_Adjust1(){
8902       SG_Adjust();
8903       /sampling = 0;
8904       sampling = 20;
8905       /sampling = 200;
8906       window.setTimeout(SG_Adjust1,sampling);
8907     }
8908   
```

```

8910 }
8911 function SightGlass_Setup(){
8912   SG_Adjust();
8913   window.addEventListener('resize',SG_Adjust);
8914   window.addEventListener('mousemove',SG_Adjust);
8915   window.addEventListener('scroll',SG_Adjust);
8916   window.addEventListener('wheel',SG_Wheel,{passive:false});
8917   //window.moveTo(0,0)
8918
8919   // if focused
8920   //  window.setInterval(SG_Adjust,1);
8921   //  window.setTimeout(SG_Adjust,1);
8922 }
8923 // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
8924 function ElectronSetup(){
8925   const BrowserWindow = require('electron');
8926   const currentWindow = BrowserWindow.getFocusedWindow();
8927   //currentWindow.on('move',function(){ SG_Adjust(); });
8928   currentWindow.addEventListener('move',SG_Adjust);
8929 }
8930 </script>
8931
8932 <input id="SightGlass_WorkCodeViewOpen" class="HTMLCodeViewButton" type="button" value="ShowCode">
8933 <input id="SightGlass_WorkOpenSnapshot" class="HTMLCodeViewButton" type="button" value="Snapshot">
8934 <input id="SightGlass_WorkCodeSignature" class="HTMLCodeViewButton" type="button" value="Signature">
8935 <span id="SightGlass_WorkCodeView"></span>
8936 <script id="SightGlass_WorkScript">
8937   function SightGlass_showWorkCode(){
8938     function SightGlass_showWorkCode(){
8939       showHtmlCode(SightGlass_WorkCodeSpan);
8940     }
8941     SightGlass_WorkCodeViewOpen.addEventListener('click',SightGlass_showWorkCode);
8942   }
8943   SightGlass_OpenWorkCodeView(); // should be invoked by an event
8944 </script>
8945 </details>
8946 <!-- SightGlass_WorkCodeSpan -->
8947 </!-->
8948 <!-- ===== Work -->
8949
8950
8951
8952 /**
8953 <-- ----- GJConsole BEGIN ( ----- -->
8954 <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
8955 <details><summary>GJ Console</summary>
8956 <p>
8957   <span id="GJE_RootNode0"></span>
8958   <span id="GJC_Container"></span>
8959   <style id="GJConsoleStyle">
8960     .GJConsole {
8961       z-index:1000;
8962       width:400; height:200px;
8963       margin:1px;
8964       color:#fff; background-color:#66a;
8965       font-size:12px; font-family:monospace,Courier New;
8966     }
8967   </style>
8968 </style>
8969
8970 <script id="GJConsoleScript" class="GJConsole">
8971   var GJC = {};
8972   function GJC_KeyDown(keyevent){
8973     key = keyevent.code;
8974     if( key == "Enter" ){
8975       GJC.Command(this);
8976       this.value += "\n" + FS1 // prompt
8977     }else if( key == "Escape" ){
8978       SuppressGJShell = false
8979       GhhMenu.focus() // should be previous focus
8980     }
8981   }
8982   var GJC_SessionId;
8983   function GJC_SetSessionId(){
8984     var xd = new Date();
8985     GJC_SessionId = xd.getTime() / 1000
8986   }
8987   GJC_SetSessionId();
8988   function GJC_Memory(mem,args,text){
8989     argv = args.split(' ')
8990     cmd = argv[0]
8991     argv.shift()
8992     args = argv.join(' ')
8993     ret = '';
8994
8995     if( cmd == 'clear' ){
8996       Permanent.setItem(mem,'')
8997     }else if( cmd == 'read' ){
8998       ret = Permanent.getItem(mem)
8999     }else if( cmd == 'save' ){
9000       val = Permanent.getItem(mem)
9001       if( val == null ) { val = "" }
9002       d = new Date();
9003       val += d.getTime() / 10000 + "GJC_SessionId" + "document.URL" + args + "\n"
9004       val += textValue
9005       Permanent.setItem(mem,val)
9006     }else{
9007       ret = "Commands: write | read | save | clear"
9008     }
9009   }
9010   if( cmd == 'write' ){
9011     val = Permanent.getItem(mem)
9012     if( val == null ) { val = "" }
9013     d = new Date();
9014     val += d.getTime() / 10000 + "GJC_SessionId" + "document.URL" + args + "\n"
9015     Permanent.setItem(mem,val)
9016   }else{
9017     ret = "Commands: write | read | save | clear"
9018   }
9019   return ret
9020 }
9021 // -- 2020-09-14 added TableEditor
9022 var GJE_CurElement = null; //GJE_RootNode
9023 var GJE_CaretColor = null;
9024 GJE_TableNo = 1
9025 function GJE_StyleKeyCommand(kev){
9026   keycode = kev.code
9027   console.log('GJE-Key: '+keycode)
9028   if( keycode == 'Escape' ){
9029     GJE_SetStyle(this);
9030   }
9031   kev.stopPropagation()
9032   // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
9033 }
9034 var GJE_CommandMode = false
9035 function GJE_TableKeyCommand(kev,tab){
9036   wasCmdMode = GJE_CommandMode
9037   key = kev.code
9038   if( key == 'Escape' ){
9039     console.log('To command mode: '+tab.nodeName+"#"+tab.id)
9040     //tab.setAttribute("contenteditable","false")
9041     tab.style.caretColor = "blue"
9042     GJE_CommandMode = true
9043   }else{
9044     if( key == "KeyA" ){
9045       tab.style.caretColor = "red"
9046       GJE_CommandMode = false
9047     }else{
9048       if( key == "KeyV" ){
9049         tab.style.caretColor = "red"
9050         GJE_CommandMode = false
9051       }else if( key == "KeyO" ){
9052         tab.style.caretColor = "red"
9053         GJE_CommandMode = false
9054       }else if( key == "KeyJ" ){
9055         console.log("ROW-DOWN")
9056       }else if( key == "KeyK" ){
9057         console.log("ROW-UP")
9058       }else if( key == "KeyW" ){
9059         console.log("COL-FORW")
9060       }else if( key == "KeyB" ){
9061         console.log("COL-BACK")
9062       }
9063     }
9064   }
9065   kev.stopPropagation()
9066   if( wasCmdMode ){
9067     kev.preventDefault()
9068   }
9069 }

```

```

9072     }
9073   }
9074   function GJE_DragEvent(ev,elem){
9075     x = ev.clientX
9076     y = ev.clientY
9077     console.log("Dragged: "+this.nodeName+'#'+this.id+' x='+x+ ' y=' +y)
9078   }
9079   // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
9080   // https://www.w3.org/TR/uievents/#events-mouseevents
9081   function GJE_DropEvent(ev,elem){
9082     x = ev.clientX
9083     y = ev.clientY
9084     this.style.x = x
9085     this.style.y = y
9086     this.style.position = 'absolute' // 'fixed'
9087     this.parentNode = gsh // just for test
9088     console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+ ' y=' +y)
9089     + parent"+this.parentNode.id)
9090   }
9091   function GJE_SetTableStyle(ev){
9092     this.innerHTML = this.value; // sync. for external representation?
9093     if(false){
9094       stdid = this.parentNode.id+this.id
9095       // and remove '_span' at the end
9096       e = document.getElementById(stdid)
9097       //alert('SetTableStyle #' +e.id+'\n'+this.value)
9098       if(e != null){
9099         e.innerHTML = this.value
9100       }else{
9101         console.log('Style Not found: '+stdid)
9102       }
9103     }
9104     //alert('event StopPropagation: '+ev)
9105   }
9106   function setCSSofClass(cclass,cstyle){
9107     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
9108     rlen = ss.cssRules.length;
9109     let tabrule = null;
9110     rulex = -1;
9111     // should skip white space at the top of cstyle
9112     sel = cstyle.charAt(0);
9113     selector = sel+cclass;
9114     console.log("- search style rule for '"+selector)
9115     for(let i = 0; i < rlen; i++){
9116       cr = ss.cssRules[i];
9117       console.log('CSS rule ['+i+'/'+rlen+'] '+cr.selectorText);
9118       if( cr.selectorText == selector){ // css class selector
9119         console.log('CSS rule found for:['+i+'/'+rlen+'] '+selector);
9120         ss.deleteRule(i);
9121         //rlen = ss.cssRules.length;
9122         rulex = i;
9123       }
9124       // should search and replace the property here
9125     }
9126     // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
9127     if( tabrule == null ){
9128       console.log('CSS rule NOT found for:['+rlen+'] '+selector);
9129       ss.insertRule(cclass,rlen); // override by 0
9130       console.log('CSS rule inserted:['+(rlen+1)+']\n'+cstyle);
9131     }else{
9132       ss.insertRule(cclass,rlen);
9133       ss.insertRule(cclass,0);
9134       console.log('CSS rule replaced:['+(rlen+1)+']\n'+cstyle);
9135     }
9136   }
9137   function GJE_SetStyle(te){
9138     console.log('Apply the style to:' +te.id+'\n');
9139     console.log('Apply the style to:' +te.parentNode.id+'\n');
9140     console.log('Apply the style to:' +te.parentNode.className+'\n');
9141     cclass = te.parentNode.className;
9142     setCSSofClass(cclass,te.value); // should get selector part from
9143     // selector { rules }
9144   }
9145   if(false){
9146     //console.log('Apply the style:')
9147     //stdid = this.parentNode.id+this.id+
9148     //stdid = this.id+'.style'
9149     css = te.value
9150     stdid = te.parentNode.id+'.style'
9151     e = document.getElementById(stdid);
9152     if( e != null ){
9153       //console.log('Apply the style:'+e.id+'\n'+te.value);
9154       e.innerHTML = css; // te.value;
9155       //ncss = e.sheet;
9156       //ncss.insertRule(te.value,ncss.cssRules.length);
9157     }else{
9158       console.log('No element to Apply the style: '+stdid)
9159     }
9160     tblid = te.parentNode.id+'.table';
9161     e = document.getElementById(tblkid);
9162     if( e != null ){
9163       e.setAttribute('style',css);
9164       e.setProperty('style',css,'!important');
9165     }
9166   }
9167   function makeTable(argv){
9168     //tid = ''
9169     //cwe = GJE_CurElement
9170     cwe = GJC1_Container;
9171     //cwd = GJFactory;
9172     tid = 'table_'+ GJE_TableNo
9173     nt = new Text('\n')
9174     cwe.appendChild(nt)
9175     ne = document.createElement('span'); // the container
9176     ne.appendChild(ne);
9177     ne.id = tid + '-span'
9178     ne.setAttribute('contenteditable',true)
9179     hspan = document.createElement('span'); // html part
9180     hspan.id = tid + '-html'
9181     //ne.innerHTML = '\n'
9182     nt = new Text('\n')
9183     nt.appendChild(nt)
9184     nt.appendChild(hspan)
9185     ne.setAttribute('contenteditable',true)
9186     ne.setgetAttribute('draggable',true)
9187     ne.addEventListener('drag',GJE_DragEvent);
9188     ne.addEventListener('dragend',GJE_DropEvent);
9189     var col = 3
9190     var row = 2
9191     if( argv[0] != null ){
9192       col = argv[0]
9193       argv.shift()
9194     }
9195     if( argv[0] != null ){
9196       row = argv[0]
9197       argv.shift()
9198     }
9199     //ne.setAttribute('class',tid)
9200     //nt += '<'+table + ' id="'+tid+'"' + ' class="'+tid+'"' +
9201     //nt += '<'+table + ' ondragstart="GJE_DragEvent(event,this)"' +
9202     //nt += '<'+table + ' ondragend="GJE_DropEvent(event,this)"\n' +
9203     //nt += ' dragable="true"\n' +
9204     //nt += ' contenteditable="true"' +
9205     //nt += '\n'
9206     nt += '<'+tbody + '\n';
9207     for( r=0; r<row; r++ ){
9208       nt += '<'+tr + '\n';
9209       for( c = 0; c < col; c++ ){
9210         nt += '<'+td + '\n';
9211         ht += "ABCDEFIGHIJKLMNOPQRSTUVWXYZ.charAt(c) + r
9212         ht += '<'+td + '\n';
9213       }
9214       ht += "<"+tr + '\n';
9215     }
9216     ht += '<'+tbody + '\n';
9217   }

```

```

9234 ht += '<'+table+'>\n';
9235 hspan.innerHTML = ht;
9236 nt = new Text('\n')
9237 ne.appendChild(nt)
9238 st = '#'+tid' *{(\n' // # for instance specific
9239 + "#'+border:1px solid #aaa;\n"
9240 + "#'+background-color:#fff;\n"
9241 + "#'+color:#222;\n"
9242 + "#'+font-size:#1pt !important;\n"
9243 + "#'+font-family:monospace,Courier New !important;\n"
9245 +') /* hit ESC to apply */+\n'
9246
9247 // wish script to be included
9248 //nj = document.createElement('script')
9249 //nj.appendChild(nj)
9250 //nj.innerHTML = 'function SetStyle(e){'
9251
9252 // selector seems lost in dynamic style appending
9253 if(false){
9254 ns = document.createElement('style')
9255 ns.appendChild(ns)
9256 ns.id = tid + '_style'
9257 ns.innerHTML = '\n'+st
9258 nt = new Text('\n')
9259 ne.appendChild(nt)
9260
9261 setCSSofClass(tid,st); // should be in JavaScript script?
9262
9263 ns = document.createElement('textarea')
9264 ns.appendChild(ns)
9265 ns.id = tid + '_table_def'
9266 ns.setAttribute('class','GJ_StyleEditor')
9267 ns.spellcheck = false
9268 ns.cols = 60
9269 ns.rows = 10
9270 ns.innerHTML = '\n'+st
9271 nx.addEventListener('change',GJE_SetTableStyle);
9272 nx.addEventListener('keydown',GJE_StyleKeyCommand);
9273 //nx.addEventListener('click',GJE_SetTableStyle);
9274
9275 ns = new Text('\n')
9276 cwe.appendChild(ns)
9277
9278 GJE_TableNo += 1
9279 return 'created TABLE id="'+tid+'"' ;
9280 }
9281 function GJE_NodeEdit(argv){
9282 cwe = cwe.CurElement
9283 cmd = argv[0]
9284 argv.shift()
9285 args = argv.join(' ')
9286 ret = ""
9287
9288 if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
9289 if( GJE_NodeSaved != null ){
9290 xn = GJE_RootNode
9291 GJE_NodeSaved = xn
9292 ret = "- did undo"
9293 }else{
9294 ret = '-- could not undo'
9295 }
9296 return ret
9297 }
9298 GJE_NodeSaved = GJE_RootNode.cloneNode()
9299 if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
9300 if( argv[0] == null ){
9301 ne = GJE_RootNode
9302 }else
9303 if( argv[0] == '.' ){
9304 ne = cwe.parentNode
9305 }else
9306 ne = document.getElementById(argv[0])
9307 if( ne != null ){
9308 GJE_CurElement = ne
9309 ret = "-- current node: " + ne.id
9310 }else
9311 ret = "-- not found: " + argv[0]
9312 }
9313 else
9314 if( cmd == '.mk' || cmd == '.mktable' ){
9315 makeTable(argv)
9316 }else
9317 if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
9318 ne = document.createElement(argv[0])
9319 //ne.id = argv[0]
9320 ret = "- created " + ne + " under " + cwe.tagName + "#" + cwe.id
9321 cwe.appendChild(ne)
9322 if( cmd == '.m' || cmd == '.mk' ){
9323 GJE_CurElement = ne
9324 }
9325 else
9326 if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
9327 if( cwe.id = argv[0] )
9328 if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
9329 if( cmd == '.h' || cmd == '.sh' || cmd == 'sh' ){
9330 args = argv.join('')
9331 cwe.innerHTML = s
9332 else
9333 if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
9334 cwe.setAttribute(argv[0],argv[1])
9335 }else
9336 if( cmd == '.l' ){
9337 if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
9338 ret = cwe.innerHTML
9339 }else
9340 if( cmd == '.pw' || cmd == 'pw' || cmd == 'pw' ){
9341 ret = cwe.getAttribute('name') + " " + cwe.id
9342 for( we = cwe.parentNode; we != null; ){
9343 ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
9344 we = we.parentNode
9345 }
9346 }
9347 else
9348 if( cmd == 'mk | rm \n' ){
9349 ret += " mk type -- make node with name and type\n"
9350 for( we = cwe.parentNode; we != null; ){
9351 ret += " nm name -- set the id #name of current node\n"
9352 we = we.parentNode
9353 }
9354 }
9355 }
9356 }
9357 }
9358 }
9359 }
9360 }
9361 }
9362 }
9363 }
9364 function GJC_Command(text){
9365 lines = text.value.split('\n')
9366 line = lines[lines.length-1]
9367 argv = line.split(' ')
9368 text.value += '\n'
9369 if( argv[0] == '' ){ argv.shift() }
9370 argv = argv.join(' ')
9371 cmd = argv[0]
9372 argv.shift()
9373 argv = argv.join(' ')
9374
9375 if( cmd == 'nolog' ){
9376 StopConsoleLog = true
9377 }else
9378 if( cmd == 'new' ){
9379 if( argv[0] == 'table' ){
9380 argv.shift();
9381 console.log(argv.join())
9382 text.value += makeTable(argv)
9383 }else
9384 if( argv[0] == 'console' ){
9385 text.value += GJ_NewConsole('GJ_Console')
9386 }else{
9387 text.value += '-- new [ console | table ] '
9388 }
9389 }else
9390 if( cmd == 'strip' ){
9391 //text.value += GJF_StripClass()
9392 }else
9393 if( cmd == 'css' ){
9394 sel = '#table_1';
9395 if(argv[0]==''0')
9396 rule1 = sel+'{color:#000 !important; background-color:#fff !important;}';
9397 }

```

```

9396 else
9397   rule1 = sel+'{color:#000 !important; background-color:#eef !important;}' ;
9398   document.styleSheets[3].deleteRule(0);
9399   document.styleSheets[3].insertRule(rule1,0);
9400   text.value += "CSS rule added: "+rule1
9401 }else
9402 if( cmd == 'print' ){
9403   e = null;
9404   if( e == null ){
9405     e = document.getElementById('GJFactory_0')
9406   }
9407   if( e == null ){
9408     e = document.getElementById('GJFactory_1')
9409   }
9410   if( argv[0] != null ){
9411     id = argv[0]
9412   if( id == 'f' ){
9413     //e = document.getElementById('GJE_RootNode');
9414   }else{
9415     e = document.getElementById(id)
9416   }
9417   if( e != null ){
9418     text.value += e.outerHTML
9419   }else{
9420     text.value += "Not found: " + id
9421   }
9422 }else{
9423   text.value += GJE_RootNode.outerHTML
9424   //text.value += e.innerHTML
9425 }
9426 }else
9427 if( cmd == 'destroy' ){
9428   text.value += GJFactory_Destroy()
9429 }else
9430 if( cmd == 'save' ){
9431   e = document.getElementById('GJFactory-1').innerHTML
9432   Permanent.setItem('GJFactory-1',e.innerHTML)
9433   text.value += " --- Saved GJFactory"
9434 }else
9435 if( cmd == 'load' ){
9436   gif = Permanent.getItem('GJFactory-1')
9437   e = document.getElementById('GJFactory')
9438   e.innerHTML = gif
9439   //Event listener restored
9440   text.value += " --- Eventlistener was restored"
9441 }else
9442 if( cmd.charAt(0) == '.' ){
9443   argv0 = argv0.split('.')
9444   text.value += GJE_NodeEdit(argv0)
9445 }else
9446 if( cmd == 'cont' ){
9447   bannerIsStopping = false
9448   GshMenuStop.innerHTML = "Stop"
9449 }else
9450 if( cmd == 'date' ){
9451   text.value += DateLong()
9452 }else
9453 if( cmd == 'echo' ){
9454   text.value += args
9455 }else
9456 if( cmd == 'fork' ){
9457   html_fork()
9458 }else
9459 if( cmd == 'last' ){
9460   text.value += MyHistory
9461   //e.innerHTML = document.createElement("span")
9462   //e.innerHTML = MyHistory
9463   //tx = MyHistory.replace("\n","");
9464   //text.value += tx.replace("<"+br>","\\n") + "xxxx<"+br>yyyy"
9465 }else
9466 if( cmd == 'ne' ){
9467   text.value += GJE_NodeEdit(argv)
9468 }else
9469 if( cmd == 'reload' ){
9470   location.reload()
9471 }else
9472 if( cmd == 'mem' ){
9473   text.value += GJC_Memory('GJC_Storage',args,text)
9474 }else
9475 if( cmd == 'stop' ){
9476   bannerIsStopping = true
9477   GshMenuStop.innerHTML = "Start"
9478 }else
9479 if( cmd == 'who' ){
9480   text.value += "SessionId=" + GJC_SessionId+" "+document.URL
9481 }else
9482 if( cmd == 'wall' ){
9483   text.value += GJC_Memory('GJC_Wall','write',text)
9484 }else
9485 {
9486   text.value += "Commands: help | echo | date | last \n"
9487   + '          new | save | load | mem \n'
9488   + '          who | wall | fork | nife'
9489 }
9490 }
9491 }
9492 function GJC_Input(){
9493   if( this.value.endsWith("\n") ){ // remove NL added by textarea
9494     this.value = this.value.slice(0,this.value.length-1)
9495   }
9496 }
9497 }
9498 var GJC_Id = null
9499 function GJC_Resize(){
9500   GJC_Id.style.width = 2000
9501   GJC_Id.style.height = window.innerWidth - 16
9502   GJC_Id.style.width = '100%';
9503   GJC_Id.style.height = 300;
9504   GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
9505   GJC_Id.style.color = "rgba(255,255,255,1.0)"
9506 }
9507 function GJC_FocusIn(){
9508   this.spellcheck = false
9509   SuppressGShell = true
9510   this.onkeydown = GJC_KeyDown
9511   GJC_Resize()
9512 }
9513 function GJC_FocusOut(){
9514   SuppressGShell = false
9515   this.removeEventListener('keydown',GJC_KeyDown);
9516 }
9517 window.addEventListener('resize',GJC_Resize);
9518
9519 function GJC_Onstorage(e){
9520   //alert('Got Message')
9521   //GJC.value += "\n((ReceivedMessage))\n"
9522 }
9523 window.addEventListener('storage',GJC_Onstorage);
9524 //window.addEventListener('storage',()=>alert('GotMessage'))
9525
9526 function GJC_Setup(gjcid){
9527   /gjcid.style.width = gsh.getBoundingClientRect().width
9528   gjcid.style.width = '100%';
9529   gjcid.style.value = "GShell Console // " + GshVersion.innerHTML + "\n"
9530   /gjcid.value = "Date: " + DateLong() + "\n"
9531   gjcid.value += PS1
9532   gjcid.onfocus = GJC_FocusIn
9533   gjcid.addEventListener('input',GJC_Input);
9534   gjcid.addEventListener('focusout',GJC_FocusOut);
9535   GJC_Id = gjcid
9536 }
9537 function GJC_Clear(id){
9538 }
9539 function GJConsole_initConsole(){
9540   if( document.getElementById('GJC_0') != null ){
9541     GJC_Setup(GJC_0)
9542   }else{
9543     GJC_Container.innerHTML = '<' +
9544       '+textarea id="GJC_1" class="GJConsole"><'+/textarea>';
9545     GJC_Setup(GJC_1)
9546     factor = document.createElement('span');
9547     gsh.appendChild(factor)
9548     GJE_RootNode = factory
9549     GJC_CurElement = GJE_RootNode;
9550   }
9551 }
9552 var initGJCF = false;
9553 function GJConsole_initFactory(){
9554   if( initGJCF ){ return; } initGJCF = true;
9555   GShell_initKeyCommand();
9556   GJConsole_initConsole();
9557 }
```

```

9558     }
9559     //GJConsole_initFactory();
9560     //ooo: focus handling
9561     </script>
9562     <style>
9563     .GJ_StyleEditor {
9564         font-size:9pt !important;
9565         font-family:Courier New, monospace !important;
9566     }
9567     </style>
9568
9569     <details>
9570     <span>
9571     <!-- ----- GJConsole END } ----- -->
9572     </span>
9573
9574     /*
9575     <span id="BlinderText">
9576     <style id="BlinderTextStyle">
9577     #GJConsole {
9578         position: absolute; z-index:5000;
9579         position: relative;
9580         display: block;
9581         left:9px;
9582         color:#fff;
9583         width:800px; height:300px; resize:both;
9584         margin:10px; padding:4px;
9585         background-color:rgba(200,200,200,0.5) !important;
9586     }
9587     .MsgText {
9588         width:578px !important;
9589         resize:both !important;
9590         color:#000 !important;
9591     }
9592     .CjNote {
9593         font-family:Georgia !important;
9594         font-size:13pt !important;
9595         color:#22a !important;
9596     }
9597     .textfield {
9598         display:inline;
9599         border:0.5px solid #444;
9600         border-radius:2px;
9601         color:#000; background-color:fff;
9602         width:106px; height:18px;
9603         margin:2px;
9604         padding:2px;
9605         resize:none;
9606         vertical-align:middle;
9607         font-size:10pt; font-family:Courier New;
9608     }
9609     .textLabel {
9610         border:0px solid #000 !important;
9611         background-color:rgba(0,0,0,0);
9612     }
9613     .textURL {
9614         width:300px !important;
9615         border:0px solid #000 !important;
9616         background-color:rgba(0,0,0,0);
9617     }
9618     .VisibleText {
9619     }
9620     .BlinderText {
9621         color:#000; background-color:#eee;
9622     }
9623     .joinbutton {
9624         font-family:Georgia !important;
9625         font-size:11pt;
9626         line-height:1.1;
9627         height:18px;
9628         width:50px;
9629         padding:0 !important;
9630         text-align:center !important;
9631         border-color:#aaa !important;
9632         border-radius:5px;
9633         color:#fff; background-color:#4a4 !important;
9634         vertical-align:middle !important;
9635     }
9636     .Sendbutton {
9637         vertical-align:top;
9638     }
9639     .ws0_log {
9640         font-size:10pt;
9641         color:#000 !important;
9642         line-height:1.0;
9643         background-color:rgba(255,255,255,0.7) !important;
9644         font-family:Courier New,monospace !important;
9645         width:99.3%;
9646         white-space:pre;
9647     }
9648 </style>
9649
9650 <!-- Form autofill test
9651 <input type="text" value="https://192.168.10.1/boafm/formLogin" size="80">
9652 <form method="POST" id="xform" action="https://192.168.10.1/boafm/formLogin">
9653     <input id="X0" name="dest" type="text" size="80" value="/index_contents.html">
9654 -->
9655 <details><summary>Form Auto. Filling</summary>
9656 <style>
9657     .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9658         display:inline !important; font-size:10pt !important; padding:1px !important;
9659     }
9660 </style>
9661 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
9662     <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
9663     Location:<input type="text" value="https://192.168.10.1/" size="80">
9664     Xserv:<input type="text" value="192.168.10.1" size="80" onfocus="this.select();"/>
9665     Password:<input type="password" class="xxinput" name="pass" type="password" autocomplete="on">
9666     SessionId:<input id="xxsid" class="xxinput" name="SESSION_ID" type="text" size="80">
9667     <input id="xxsub" class="xxinput" type="submit" value="Submit"></form>
9668 </span>
9669 <script>
9700     function XXSetFormAction(){
9701         xform.setAttribute('action',xxserv.value);
9702     }
9703     xform.setAttribute('action',xxserv.value);
9704     xxserv.addEventListener('change',XXSetFormAction);
9705     //xxserv.value = location.href;
9706     </script>
9707 </details>
9708 /*
9709 */
9710 <details id="blinderTextClass"><summary>BlinderText</summary>
9711 <span class="gsh-src">
9712     <span id="BlinderTextScript">
9713         // https://w3c.github.io/uievents/#event-type-keydown
9714         // 2020-09-21 class BlinderText - textarea element not to be readable
9715         // BlinderText attributes
9716         // bl_plainText - null
9717         // bl_hideChecksum - [false]
9718         // bl_showLength - [false]
9719         // bl_visible - [false]
9720         // domConfig - []
9721         // - min. length
9722         // - max. length
9723         // - acceptable charset in generete text
9724         function BlinderChecksum(text){
9725             plain = text.bl_plainText;
9726             return strCRC32(plain,plain.length).toFixed(0);
9727         }
9728         function BlinderKeydown(ev){
9729             bl_plainText.target.value = '';
9730             if(ev.code == 'Backspace'){
9731                 if(ev.code == 'Backspace'){
9732                     blnd.bl_plainText = blnd.bl_plainText.slice(0,blnd.bl_plainText.length-1)
9733                 }else{
9734                     if(andev.code == 'KeyV', ev.ctrlKey){
9735                         blnd.bl_visible = !blnd.bl_visible;
9736                     }
9737                     if(andev.code == 'KeyL', ev.ctrlKey) {
9738                         blnd.bl_showLength = !blnd.bl_showLength;
9739                     }
9740                 }
9741             }
9742         }
9743     </span>
9744 </span>
9745 </details>
9746 <script>
9747     blnd = document.getElementById('BlinderText');
9748     blnd.bl_plainText = '';
9749     blnd.bl_hideChecksum = false;
9750     blnd.bl_showLength = false;
9751     blnd.bl_visible = false;
9752     blnd.bl_domConfig = [];
9753     blnd.bl_minLength = 0;
9754     blnd.bl_maxLength = 1000;
9755     blnd.bl_acceptableCharSet = 'utf-8';
9756     blnd.bl_ctrlKey = false;
9757     blnd.bl_showLength = true;
9758     blnd.bl_hideChecksum = false;
9759     blnd.bl_minLength = 0;
9760     blnd.bl_maxLength = 1000;
9761     blnd.bl_acceptableCharSet = 'utf-8';
9762     blnd.bl_ctrlKey = false;
9763     blnd.bl_showLength = true;
9764     blnd.bl_hideChecksum = false;
9765     blnd.bl_minLength = 0;
9766     blnd.bl_maxLength = 1000;
9767     blnd.bl_acceptableCharSet = 'utf-8';
9768     blnd.bl_ctrlKey = false;
9769     blnd.bl_showLength = true;
9770     blnd.bl_hideChecksum = false;
9771     blnd.bl_minLength = 0;
9772     blnd.bl_maxLength = 1000;
9773     blnd.bl_acceptableCharSet = 'utf-8';
9774     blnd.bl_ctrlKey = false;
9775     blnd.bl_showLength = true;
9776     blnd.bl_hideChecksum = false;
9777     blnd.bl_minLength = 0;
9778     blnd.bl_maxLength = 1000;
9779     blnd.bl_acceptableCharSet = 'utf-8';
9780     blnd.bl_ctrlKey = false;
9781     blnd.bl_showLength = true;
9782     blnd.bl_hideChecksum = false;
9783     blnd.bl_minLength = 0;
9784     blnd.bl_maxLength = 1000;
9785     blnd.bl_acceptableCharSet = 'utf-8';
9786     blnd.bl_ctrlKey = false;
9787     blnd.bl_showLength = true;
9788     blnd.bl_hideChecksum = false;
9789     blnd.bl_minLength = 0;
9790     blnd.bl_maxLength = 1000;
9791     blnd.bl_acceptableCharSet = 'utf-8';
9792     blnd.bl_ctrlKey = false;
9793     blnd.bl_showLength = true;
9794     blnd.bl_hideChecksum = false;
9795     blnd.bl_minLength = 0;
9796     blnd.bl_maxLength = 1000;
9797     blnd.bl_acceptableCharSet = 'utf-8';
9798     blnd.bl_ctrlKey = false;
9799     blnd.bl_showLength = true;
9800     blnd.bl_hideChecksum = false;
9801     blnd.bl_minLength = 0;
9802     blnd.bl_maxLength = 1000;
9803     blnd.bl_acceptableCharSet = 'utf-8';
9804     blnd.bl_ctrlKey = false;
9805     blnd.bl_showLength = true;
9806     blnd.bl_hideChecksum = false;
9807     blnd.bl_minLength = 0;
9808     blnd.bl_maxLength = 1000;
9809     blnd.bl_acceptableCharSet = 'utf-8';
9810     blnd.bl_ctrlKey = false;
9811     blnd.bl_showLength = true;
9812     blnd.bl_hideChecksum = false;
9813     blnd.bl_minLength = 0;
9814     blnd.bl_maxLength = 1000;
9815     blnd.bl_acceptableCharSet = 'utf-8';
9816     blnd.bl_ctrlKey = false;
9817     blnd.bl_showLength = true;
9818     blnd.bl_hideChecksum = false;
9819     blnd.bl_minLength = 0;
9820     blnd.bl_maxLength = 1000;
9821     blnd.bl_acceptableCharSet = 'utf-8';
9822     blnd.bl_ctrlKey = false;
9823     blnd.bl_showLength = true;
9824     blnd.bl_hideChecksum = false;
9825     blnd.bl_minLength = 0;
9826     blnd.bl_maxLength = 1000;
9827     blnd.bl_acceptableCharSet = 'utf-8';
9828     blnd.bl_ctrlKey = false;
9829     blnd.bl_showLength = true;
9830     blnd.bl_hideChecksum = false;
9831     blnd.bl_minLength = 0;
9832     blnd.bl_maxLength = 1000;
9833     blnd.bl_acceptableCharSet = 'utf-8';
9834     blnd.bl_ctrlKey = false;
9835     blnd.bl_showLength = true;
9836     blnd.bl_hideChecksum = false;
9837     blnd.bl_minLength = 0;
9838     blnd.bl_maxLength = 1000;
9839     blnd.bl_acceptableCharSet = 'utf-8';
9840     blnd.bl_ctrlKey = false;
9841     blnd.bl_showLength = true;
9842     blnd.bl_hideChecksum = false;
9843     blnd.bl_minLength = 0;
9844     blnd.bl_maxLength = 1000;
9845     blnd.bl_acceptableCharSet = 'utf-8';
9846     blnd.bl_ctrlKey = false;
9847     blnd.bl_showLength = true;
9848     blnd.bl_hideChecksum = false;
9849     blnd.bl_minLength = 0;
9850     blnd.bl_maxLength = 1000;
9851     blnd.bl_acceptableCharSet = 'utf-8';
9852     blnd.bl_ctrlKey = false;
9853     blnd.bl_showLength = true;
9854     blnd.bl_hideChecksum = false;
9855     blnd.bl_minLength = 0;
9856     blnd.bl_maxLength = 1000;
9857     blnd.bl_acceptableCharSet = 'utf-8';
9858     blnd.bl_ctrlKey = false;
9859     blnd.bl_showLength = true;
9860     blnd.bl_hideChecksum = false;
9861     blnd.bl_minLength = 0;
9862     blnd.bl_maxLength = 1000;
9863     blnd.bl_acceptableCharSet = 'utf-8';
9864     blnd.bl_ctrlKey = false;
9865     blnd.bl_showLength = true;
9866     blnd.bl_hideChecksum = false;
9867     blnd.bl_minLength = 0;
9868     blnd.bl_maxLength = 1000;
9869     blnd.bl_acceptableCharSet = 'utf-8';
9870     blnd.bl_ctrlKey = false;
9871     blnd.bl_showLength = true;
9872     blnd.bl_hideChecksum = false;
9873     blnd.bl_minLength = 0;
9874     blnd.bl_maxLength = 1000;
9875     blnd.bl_acceptableCharSet = 'utf-8';
9876     blnd.bl_ctrlKey = false;
9877     blnd.bl_showLength = true;
9878     blnd.bl_hideChecksum = false;
9879     blnd.bl_minLength = 0;
9880     blnd.bl_maxLength = 1000;
9881     blnd.bl_acceptableCharSet = 'utf-8';
9882     blnd.bl_ctrlKey = false;
9883     blnd.bl_showLength = true;
9884     blnd.bl_hideChecksum = false;
9885     blnd.bl_minLength = 0;
9886     blnd.bl_maxLength = 1000;
9887     blnd.bl_acceptableCharSet = 'utf-8';
9888     blnd.bl_ctrlKey = false;
9889     blnd.bl_showLength = true;
9890     blnd.bl_hideChecksum = false;
9891     blnd.bl_minLength = 0;
9892     blnd.bl_maxLength = 1000;
9893     blnd.bl_acceptableCharSet = 'utf-8';
9894     blnd.bl_ctrlKey = false;
9895     blnd.bl_showLength = true;
9896     blnd.bl_hideChecksum = false;
9897     blnd.bl_minLength = 0;
9898     blnd.bl_maxLength = 1000;
9899     blnd.bl_acceptableCharSet = 'utf-8';
9900     blnd.bl_ctrlKey = false;
9901     blnd.bl_showLength = true;
9902     blnd.bl_hideChecksum = false;
9903     blnd.bl_minLength = 0;
9904     blnd.bl_maxLength = 1000;
9905     blnd.bl_acceptableCharSet = 'utf-8';
9906     blnd.bl_ctrlKey = false;
9907     blnd.bl_showLength = true;
9908     blnd.bl_hideChecksum = false;
9909     blnd.bl_minLength = 0;
9910     blnd.bl_maxLength = 1000;
9911     blnd.bl_acceptableCharSet = 'utf-8';
9912     blnd.bl_ctrlKey = false;
9913     blnd.bl_showLength = true;
9914     blnd.bl_hideChecksum = false;
9915     blnd.bl_minLength = 0;
9916     blnd.bl_maxLength = 1000;
9917     blnd.bl_acceptableCharSet = 'utf-8';
9918     blnd.bl_ctrlKey = false;
9919     blnd.bl_showLength = true;
9920     blnd.bl_hideChecksum = false;
9921     blnd.bl_minLength = 0;
9922     blnd.bl_maxLength = 1000;
9923     blnd.bl_acceptableCharSet = 'utf-8';
9924     blnd.bl_ctrlKey = false;
9925     blnd.bl_showLength = true;
9926     blnd.bl_hideChecksum = false;
9927     blnd.bl_minLength = 0;
9928     blnd.bl_maxLength = 1000;
9929     blnd.bl_acceptableCharSet = 'utf-8';
9930     blnd.bl_ctrlKey = false;
9931     blnd.bl_showLength = true;
9932     blnd.bl_hideChecksum = false;
9933     blnd.bl_minLength = 0;
9934     blnd.bl_maxLength = 1000;
9935     blnd.bl_acceptableCharSet = 'utf-8';
9936     blnd.bl_ctrlKey = false;
9937     blnd.bl_showLength = true;
9938     blnd.bl_hideChecksum = false;
9939     blnd.bl_minLength = 0;
9940     blnd.bl_maxLength = 1000;
9941     blnd.bl_acceptableCharSet = 'utf-8';
9942     blnd.bl_ctrlKey = false;
9943     blnd.bl_showLength = true;
9944     blnd.bl_hideChecksum = false;
9945     blnd.bl_minLength = 0;
9946     blnd.bl_maxLength = 1000;
9947     blnd.bl_acceptableCharSet = 'utf-8';
9948     blnd.bl_ctrlKey = false;
9949     blnd.bl_showLength = true;
9950     blnd.bl_hideChecksum = false;
9951     blnd.bl_minLength = 0;
9952     blnd.bl_maxLength = 1000;
9953     blnd.bl_acceptableCharSet = 'utf-8';
9954     blnd.bl_ctrlKey = false;
9955     blnd.bl_showLength = true;
9956     blnd.bl_hideChecksum = false;
9957     blnd.bl_minLength = 0;
9958     blnd.bl_maxLength = 1000;
9959     blnd.bl_acceptableCharSet = 'utf-8';
9960     blnd.bl_ctrlKey = false;
9961     blnd.bl_showLength = true;
9962     blnd.bl_hideChecksum = false;
9963     blnd.bl_minLength = 0;
9964     blnd.bl_maxLength = 1000;
9965     blnd.bl_acceptableCharSet = 'utf-8';
9966     blnd.bl_ctrlKey = false;
9967     blnd.bl_showLength = true;
9968     blnd.bl_hideChecksum = false;
9969     blnd.bl_minLength = 0;
9970     blnd.bl_maxLength = 1000;
9971     blnd.bl_acceptableCharSet = 'utf-8';
9972     blnd.bl_ctrlKey = false;
9973     blnd.bl_showLength = true;
9974     blnd.bl_hideChecksum = false;
9975     blnd.bl_minLength = 0;
9976     blnd.bl_maxLength = 1000;
9977     blnd.bl_acceptableCharSet = 'utf-8';
9978     blnd.bl_ctrlKey = false;
9979     blnd.bl_showLength = true;
9980     blnd.bl_hideChecksum = false;
9981     blnd.bl_minLength = 0;
9982     blnd.bl_maxLength = 1000;
9983     blnd.bl_acceptableCharSet = 'utf-8';
9984     blnd.bl_ctrlKey = false;
9985     blnd.bl_showLength = true;
9986     blnd.bl_hideChecksum = false;
9987     blnd.bl_minLength = 0;
9988     blnd.bl_maxLength = 1000;
9989     blnd.bl_acceptableCharSet = 'utf-8';
9990     blnd.bl_ctrlKey = false;
9991     blnd.bl_showLength = true;
9992     blnd.bl_hideChecksum = false;
9993     blnd.bl_minLength = 0;
9994     blnd.bl_maxLength = 1000;
9995     blnd.bl_acceptableCharSet = 'utf-8';
9996     blnd.bl_ctrlKey = false;
9997     blnd.bl_showLength = true;
9998     blnd.bl_hideChecksum = false;
9999     blnd.bl_minLength = 0;
10000    blnd.bl_maxLength = 1000;
10001    blnd.bl_acceptableCharSet = 'utf-8';
10002    blnd.bl_ctrlKey = false;
10003    blnd.bl_showLength = true;
10004    blnd.bl_hideChecksum = false;
10005    blnd.bl_minLength = 0;
10006    blnd.bl_maxLength = 1000;
10007    blnd.bl_acceptableCharSet = 'utf-8';
10008    blnd.bl_ctrlKey = false;
10009    blnd.bl_showLength = true;
10010    blnd.bl_hideChecksum = false;
10011    blnd.bl_minLength = 0;
10012    blnd.bl_maxLength = 1000;
10013    blnd.bl_acceptableCharSet = 'utf-8';
10014    blnd.bl_ctrlKey = false;
10015    blnd.bl_showLength = true;
10016    blnd.bl_hideChecksum = false;
10017    blnd.bl_minLength = 0;
10018    blnd.bl_maxLength = 1000;
10019    blnd.bl_acceptableCharSet = 'utf-8';
10020    blnd.bl_ctrlKey = false;
10021    blnd.bl_showLength = true;
10022    blnd.bl_hideChecksum = false;
10023    blnd.bl_minLength = 0;
10024    blnd.bl_maxLength = 1000;
10025    blnd.bl_acceptableCharSet = 'utf-8';
10026    blnd.bl_ctrlKey = false;
10027    blnd.bl_showLength = true;
10028    blnd.bl_hideChecksum = false;
10029    blnd.bl_minLength = 0;
10030    blnd.bl_maxLength = 1000;
10031    blnd.bl_acceptableCharSet = 'utf-8';
10032    blnd.bl_ctrlKey = false;
10033    blnd.bl_showLength = true;
10034    blnd.bl_hideChecksum = false;
10035    blnd.bl_minLength = 0;
10036    blnd.bl_maxLength = 1000;
10037    blnd.bl_acceptableCharSet = 'utf-8';
10038    blnd.bl_ctrlKey = false;
10039    blnd.bl_showLength = true;
10040    blnd.bl_hideChecksum = false;
10041    blnd.bl_minLength = 0;
10042    blnd.bl_maxLength = 1000;
10043    blnd.bl_acceptableCharSet = 'utf-8';
10044    blnd.bl_ctrlKey = false;
10045    blnd.bl_showLength = true;
10046    blnd.bl_hideChecksum = false;
10047    blnd.bl_minLength = 0;
10048    blnd.bl_maxLength = 1000;
10049    blnd.bl_acceptableCharSet = 'utf-8';
10050    blnd.bl_ctrlKey = false;
10051    blnd.bl_showLength = true;
10052    blnd.bl_hideChecksum = false;
10053    blnd.bl_minLength = 0;
10054    blnd.bl_maxLength = 1000;
10055    blnd.bl_acceptableCharSet = 'utf-8';
10056    blnd.bl_ctrlKey = false;
10057    blnd.bl_showLength = true;
10058    blnd.bl_hideChecksum = false;
10059    blnd.bl_minLength = 0;
10060    blnd.bl_maxLength = 1000;
10061    blnd.bl_acceptableCharSet = 'utf-8';
10062    blnd.bl_ctrlKey = false;
10063    blnd.bl_showLength = true;
10064    blnd.bl_hideChecksum = false;
10065    blnd.bl_minLength = 0;
10066    blnd.bl_maxLength = 1000;
10067    blnd.bl_acceptableCharSet = 'utf-8';
10068    blnd.bl_ctrlKey = false;
10069    blnd.bl_showLength = true;
10070    blnd.bl_hideChecksum = false;
10071    blnd.bl_minLength = 0;
10072    blnd.bl_maxLength = 1000;
10073    blnd.bl_acceptableCharSet = 'utf-8';
10074    blnd.bl_ctrlKey = false;
10075    blnd.bl_showLength = true;
10076    blnd.bl_hideChecksum = false;
10077    blnd.bl_minLength = 0;
10078    blnd.bl_maxLength = 1000;
10079    blnd.bl_acceptableCharSet = 'utf-8';
10080    blnd.bl_ctrlKey = false;
10081    blnd.bl_showLength = true;
10082    blnd.bl_hideChecksum = false;
10083    blnd.bl_minLength = 0;
10084    blnd.bl_maxLength = 1000;
10085    blnd.bl_acceptableCharSet = 'utf-8';
10086    blnd.bl_ctrlKey = false;
10087    blnd.bl_showLength = true;
10088    blnd.bl_hideChecksum = false;
10089    blnd.bl_minLength = 0;
10090    blnd.bl_maxLength = 1000;
10091    blnd.bl_acceptableCharSet = 'utf-8';
10092    blnd.bl_ctrlKey = false;
10093    blnd.bl_showLength = true;
10094    blnd.bl_hideChecksum = false;
10095    blnd.bl_minLength = 0;
10096    blnd.bl_maxLength = 1000;
10097    blnd.bl_acceptableCharSet = 'utf-8';
10098    blnd.bl_ctrlKey = false;
10099    blnd.bl_showLength = true;
10100    blnd.bl_hideChecksum = false;
10101    blnd.bl_minLength = 0;
10102    blnd.bl_maxLength = 1000;
10103    blnd.bl_acceptableCharSet = 'utf-8';
10104    blnd.bl_ctrlKey = false;
10105    blnd.bl_showLength = true;
10106    blnd.bl_hideChecksum = false;
10107    blnd.bl_minLength = 0;
10108    blnd.bl_maxLength = 1000;
10109    blnd.bl_acceptableCharSet = 'utf-8';
10110    blnd.bl_ctrlKey = false;
10111    blnd.bl_showLength = true;
10112    blnd.bl_hideChecksum = false;
10113    blnd.bl_minLength = 0;
10114    blnd.bl_maxLength = 1000;
10115    blnd.bl_acceptableCharSet = 'utf-8';
10116    blnd.bl_ctrlKey = false;
10117    blnd.bl_showLength = true;
10118    blnd.bl_hideChecksum = false;
10119    blnd.bl_minLength = 0;
10120    blnd.bl_maxLength = 1000;
10121    blnd.bl_acceptableCharSet = 'utf-8';
10122    blnd.bl_ctrlKey = false;
10123    blnd.bl_showLength = true;
10124    blnd.bl_hideChecksum = false;
10125    blnd.bl_minLength = 0;
10126    blnd.bl_maxLength = 1000;
10127    blnd.bl_acceptableCharSet = 'utf-8';
10128    blnd.bl_ctrlKey = false;
10129    blnd.bl_showLength = true;
10130    blnd.bl_hideChecksum = false;
10131    blnd.bl_minLength = 0;
10132    blnd.bl_maxLength = 1000;
10133    blnd.bl_acceptableCharSet = 'utf-8';
10134    blnd.bl_ctrlKey = false;
10135    blnd.bl_showLength = true;
10136    blnd.bl_hideChecksum = false;
10137    blnd.bl_minLength = 0;
10138    blnd.bl_maxLength = 1000;
10139    blnd.bl_acceptableCharSet = 'utf-8';
10140    blnd.bl_ctrlKey = false;
10141    blnd.bl_showLength = true;
10142    blnd.bl_hideChecksum = false;
10143    blnd.bl_minLength = 0;
10144    blnd.bl_maxLength = 1000;
10145    blnd.bl_acceptableCharSet = 'utf-8';
10146    blnd.bl_ctrlKey = false;
10147    blnd.bl_showLength = true;
10148    blnd.bl_hideChecksum = false;
10149    blnd.bl_minLength = 0;
10150    blnd.bl_maxLength = 1000;
10151    blnd.bl_acceptableCharSet = 'utf-8';
10152    blnd.bl_ctrlKey = false;
10153    blnd.bl_showLength = true;
10154    blnd.bl_hideChecksum = false;
10155    blnd.bl_minLength = 0;
10156    blnd.bl_maxLength = 1000;
10157    blnd.bl_acceptableCharSet = 'utf-8';
10158    blnd.bl_ctrlKey = false;
10159    blnd.bl_showLength = true;
10160    blnd.bl_hideChecksum = false;
10161    blnd.bl_minLength = 0;
10162    blnd.bl_maxLength = 1000;
10163    blnd.bl_acceptableCharSet = 'utf-8';
10164    blnd.bl_ctrlKey = false;
10165    blnd.bl_showLength = true;
10166    blnd.bl_hideChecksum = false;
10167    blnd.bl_minLength = 0;
10168    blnd.bl_maxLength = 1000;
10169    blnd.bl_acceptableCharSet = 'utf-8';
10170    blnd.bl_ctrlKey = false;
10171    blnd.bl_showLength = true;
10172    blnd.bl_hideChecksum = false;
10173    blnd.bl_minLength = 0;
10174    blnd.bl_maxLength = 1000;
10175    blnd.bl_acceptableCharSet = 'utf-8';
10176    blnd.bl_ctrlKey = false;
10177    blnd.bl_showLength = true;
10178    blnd.bl_hideChecksum = false;
10179    blnd.bl_minLength = 0;
10180    blnd.bl_maxLength = 1000;
10181    blnd.bl_acceptableCharSet = 'utf-8';
10182    blnd.bl_ctrlKey = false;
10183    blnd.bl_showLength = true;
10184    blnd.bl_hideChecksum = false;
10185    blnd.bl_minLength = 0;
10186    blnd.bl_maxLength = 1000;
10187    blnd.bl_acceptableCharSet = 'utf-8';
10188    blnd.bl_ctrlKey = false;
10189    blnd.bl_showLength = true;
10190    blnd.bl_hideChecksum = false;
10191    blnd.bl_minLength = 0;
10192    blnd.bl_maxLength = 1000;
10193    blnd.bl_acceptableCharSet = 'utf-8';
10194    blnd.bl_ctrlKey = false;
10195    blnd.bl_showLength = true;
10196    blnd.bl_hideChecksum = false;
10197    blnd.bl_minLength = 0;
10198    blnd.bl_maxLength = 1000;
10199    blnd.bl_acceptableCharSet = 'utf-8';
10200    blnd.bl_ctrlKey = false;
10201    blnd.bl_showLength = true;
10202    blnd.bl_hideChecksum = false;
10203    blnd.bl_minLength = 0;
10204    blnd.bl_maxLength = 1000;
10205    blnd.bl_acceptableCharSet = 'utf-8';
10206    blnd.bl_ctrlKey = false;
10207    blnd.bl_showLength = true;
10208    blnd.bl_hideChecksum = false;
10209    blnd.bl_minLength = 0;
10210    blnd.bl_maxLength = 1000;
10211    blnd.bl_acceptableCharSet = 'utf-8';
10212    blnd.bl_ctrlKey = false;
10213    blnd.bl_showLength = true;
10214    blnd.bl_hideChecksum = false;
10215    blnd.bl_minLength = 0;
10216    blnd.bl_maxLength = 1000;
10217    blnd.bl_acceptableCharSet = 'utf-8';
10218    blnd.bl_ctrlKey = false;
10219    blnd.bl_showLength = true;
10220    blnd.bl_hideChecksum = false;
10221    blnd.bl_minLength = 0;
10222    blnd.bl_maxLength = 1000;
10223    blnd.bl_acceptableCharSet = 'utf-8';
10224    blnd.bl_ctrlKey = false;
10225    blnd.bl_showLength = true;
10226    blnd.bl_hideChecksum = false;
10227    blnd.bl_minLength = 0;
10228    blnd.bl_maxLength = 1000;
10229    blnd.bl_acceptableCharSet = 'utf-8';
10230    blnd.bl_ctrlKey = false;
10231    blnd.bl_showLength = true;
10232    blnd.bl_hideChecksum = false;
10233    blnd.bl_minLength = 0;
10234    blnd.bl_maxLength = 1000;
10235    blnd.bl_acceptableCharSet = 'utf-8';
10236    blnd.bl_ctrlKey = false;
10237    blnd.bl_showLength = true;
10238    blnd.bl_hideChecksum = false;
10239    blnd.bl_minLength = 0;
10240    blnd.bl_maxLength = 1000;
10241    blnd.bl_acceptableCharSet = 'utf-8';
10242    blnd.bl_ctrlKey = false;
10243    blnd.bl_showLength = true;
10244    blnd.bl_hideChecksum = false;
10245    blnd.bl_minLength = 0;
10246    blnd.bl_maxLength = 1000;
10247    blnd.bl_acceptableCharSet = 'utf-8';
10248    blnd.bl_ctrlKey = false;
10249    blnd.bl_showLength = true;
10250    blnd.bl_hideChecksum = false;
10251    blnd.bl_minLength = 0;
10252    blnd.bl_maxLength = 1000;
10253    blnd.bl_acceptableCharSet = 'utf-8';
10254    blnd.bl_ctrlKey = false;
10255    blnd.bl_showLength = true;
10256    blnd.bl_hideChecksum = false;
10257    blnd.bl_minLength = 0;
10258    blnd.bl_maxLength = 1000;
10259    blnd.bl_acceptableCharSet = 'utf-8';
10260    blnd.bl_ctrlKey = false;
10261    blnd.bl_showLength = true;
10262    blnd.bl_hideChecksum = false;
10263    blnd.bl_minLength = 0;
10264    blnd.bl_maxLength = 1000;
10265    blnd.bl_acceptableCharSet = 'utf-8';
10266    blnd.bl_ctrlKey = false;
10267    blnd.bl_showLength = true;
10268    blnd.bl_hideChecksum = false;
10269    blnd.bl_minLength = 0;
10270    blnd.bl_maxLength = 1000;
10271    blnd.bl_acceptableCharSet = 'utf-8';
10272    blnd.bl_ctrlKey = false;
10273    blnd.bl_showLength = true;
10274    blnd.bl_hideChecksum = false;
10275    blnd.bl_minLength = 0;
10276    blnd.bl_maxLength = 1000;
10277    blnd.bl_acceptableCharSet = 'utf-8';
10278    blnd.bl_ctrlKey = false;
10279    blnd.bl_showLength = true;
10280    blnd.bl_hideChecksum = false;
10281    blnd.bl_minLength = 0;
10282    blnd.bl_maxLength = 1000;
10283    blnd.bl_acceptableCharSet = 'utf-8';
10284    blnd.bl_ctrlKey = false;
10285    blnd.bl_showLength = true;
10286    blnd.bl_hideChecksum = false;
10287    blnd.bl_minLength = 0;
10288    blnd.bl_maxLength = 1000;
10289    blnd.bl_acceptableCharSet = 'utf-8';
10290    blnd.bl_ctrlKey = false;
10291    blnd.bl_showLength = true;
10292    blnd.bl_hideChecksum = false;
10293    blnd.bl_minLength = 0;
10294    blnd.bl_maxLength = 1000;
10295    blnd.bl_acceptableCharSet = 'utf-8';
10296    blnd.bl_ctrlKey = false;
10297    blnd.bl_showLength = true;
10298    blnd.bl_hideChecksum = false;
10299    blnd.bl_minLength = 0;
10300    blnd.bl_maxLength = 1000;
10301    blnd.bl_acceptableCharSet = 'utf-8';
10302    blnd.bl_ctrlKey = false;
10303    blnd.bl_showLength = true;
10304    blnd.bl_hideChecksum = false;
10305    blnd.bl_minLength = 0;
10306    blnd.bl_maxLength = 1000;
10307    blnd.bl_acceptableCharSet = 'utf-8';
10308    blnd.bl_ctrlKey = false;
10309    blnd.bl_showLength = true;
10310    blnd.bl_hideChecksum = false;
10311    blnd.bl_minLength = 0;
10312    blnd.bl_maxLength = 1000;
10313    blnd.bl_acceptableCharSet = 'utf-8';
10314    blnd.bl_ctrlKey = false;
10315    blnd.bl_showLength = true;
10316    blnd.bl_hideChecksum = false;
10317    blnd.bl_minLength = 0;
10318    blnd.bl_maxLength = 1000;
10319    blnd.bl_acceptableCharSet = 'utf-8';
10320    blnd.bl_ctrlKey = false;
10321    blnd.bl_showLength = true;
10322    blnd.bl_hideChecksum = false;
10323    blnd.bl_minLength = 0;
10324    blnd.bl_maxLength = 1000;
10325    blnd.bl_acceptableCharSet = 'utf-8';
10326    blnd.bl_ctrlKey = false;
10327    blnd.bl_showLength = true;
10328    blnd.bl_hideChecksum = false;
10329    blnd.bl
```

```

9720 if( and(ev.code == 'KeyU', ev.ctrlKey) ){
9721   blind.bl_plaintext = "";
9722 }else{
9723   if( and(ev.code == 'KeyP', ev.ctrlKey) ){
9724     checksum = BlinderChecksum(blind);
9725     blind.bl_plaintext = checksum; //toString(32);
9726   }else{
9727     if( ev.code == 'Enter' ){
9728       ev.stopPropagation();
9729       ev.preventDefault();
9730       return;
9731     }else{
9732       if( ev.key.length != 1 ){
9733         console.log('KeyUp: '+ev.code+'/'+ev.key);
9734         return;
9735       }else{
9736         blind.bl_plaintext += ev.key;
9737       }
9738     }
9739     leng = blind.bl_plaintext.length;
9740     //console.log('KeyUp: '+ev.code+'/'+blind.bl_plaintext);
9741     checksum = BlinderChecksum(blind) % 10; // show last one digit only
9742   }
9743   visual = '';
9744   if( !blind.bl_hideChecksum || blind.bl_showLength ){
9745     visual += '[';
9746   }
9747   if( !blind.bl_hideCheckSum ){
9748     visual += "#"+checksum.toString(10);
9749   }
9750   if( blind.bl_showLength ){
9751     visual += '/' + leng;
9752   }
9753   if( !blind.bl_hideCheckSum || blind.bl_showLength ){
9754     visual += "]";
9755   }
9756   if( blind.bl_visible ){
9757     visual += blind.bl_plaintext;
9758   }else{
9759     visual += '*' .repeat(leng);
9760   }
9761   blind.value = visual;
9762 }
9763 function BlinderKeyup(ev){
9764   BlinderKeyup(ev);
9765   ev.stopPropagation();
9766 }
9767 // https://w3c.github.io/uievents/#keyboardevent
9768 // https://w3c.github.io/uievents/#uievent
9769 // https://dom.spec.whatwg.org/#event
9770 function BlinderTextEvent(){
9771   ev = event;
9772   blind = ev.target;
9773   console.log('Event '+ev.type+'#'+blind.nodeName+'#'+blind.id);
9774   if( ev.type == 'keyup' ){
9775     Blinderkeyup(ev);
9776   }else{
9777     if( ev.type == 'keydown' ){
9778       Blinderkeydown(ev);
9779     }else{
9780       console.log('thru-event '+ev.type+'#'+blind.nodeName+'#'+blind.id);
9781     }
9782 }
9783 //< textarea hidden id="BlinderTextClassDef" class="textField"
9784 // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
9785 // style="width: 100%; height: 100%; border: none;">
9786 // < textarea hidden id="g1_pass1"
9787 // class="textField BlinderText"
9788 // placeholder="PassWord1"
9789 // onkeydown="BlinderTextEvent()"
9790 // onkeyup="BlinderTextEvent()"
9791 // style="width: 100%; height: 100%; border: none;">
9792 function SetupBlinderText(parent,txa,phold){
9793   if( txa == null ){
9794     txa = document.createElement('textarea');
9795     //txa.id = id;
9796   }
9797   txa.setAttribute('class','textField BlinderText');
9798   txa.setAttribute('placeholder',phold);
9799   txa.setAttribute('onkeydown','BlinderTextEvent()');
9800   txa.setAttribute('onkeyup','BlinderTextEvent()');
9801   txa.setAttribute('spellcheck','false');
9802   //txa.setAttribute('bl_plaintext','false');
9803   //txa.bl_plaintext = '';
9804   //parent.appendChild(txa);
9805 }
9806 function DestroyBlinderText(txa){
9807   txa.removeAttribute('class');
9808   txa.removeAttribute('placeholder');
9809   txa.removeAttribute('onkeydown');
9810   txa.removeAttribute('onkeyup');
9811   txa.removeAttribute('spellcheck');
9812   txa.bl_plaintext = '';
9813 }
9814 /**
9815 // visible textarea like Username
9816 */
9817 function VisibleTextEvent(){
9818   if( event.code == 'Enter' ){
9819     if( event.target.NoEnter ){
9820       event.preventDefault();
9821     }
9822   }
9823   event.stopPropagation();
9824 }
9825 function SetupVisibleText(parent,txa,phold){
9826   if( false ){
9827     txa.setAttribute('class','textField VisibleText');
9828   }else{
9829     newclass = txa.getAttribute('class');
9830     if( and(newclass != null, newclass != '') ){
9831       newclass += ' ';
9832     }
9833     newclass += 'VisibleText';
9834     txa.setAttribute('class',newclass);
9835   }
9836   //console.log('SetupVisibleText class="'+txa.class);
9837   txa.setAttribute('placeholder',phold);
9838   txa.setAttribute('onkeydown','VisibleTextEvent()');
9839   txa.setAttribute('onkeyup','VisibleTextEvent()');
9840   txa.setAttribute('spellcheck','false');
9841   cols = txa.getAttribute('cols');
9842   if( cols != null ){
9843     txa.style.width = '580px';
9844   }
9845   //console.log('VisibleText#'+txa.id+' cols='+cols)
9846 }else{
9847   //console.log('VisibleText#'+txa.id+' NO cols')
9848 }
9849 rows = txa.getAttribute('rows');
9850 if( rows != null ){
9851   txa.style.height = '30px';
9852   txa.style.resize = 'both';
9853   txa.NoEnter = false;
9854 }else{
9855   txa.NoEnter = true;
9856 }
9857 }
9858 function DestroyVisibleText(txa){
9859   txa.removeAttribute('class');
9860   txa.removeAttribute('placeholder');
9861   txa.removeAttribute('onkeydown');
9862   txa.removeAttribute('onkeyup');
9863   txa.removeAttribute('spellcheck');
9864   cols = txa.removeAttribute('cols');
9865 }
9866 </span>
9867 <script>
9868   js = document.getElementById('BlinderTextScript');
9869   eval(js.innerHTML);
9870   //js.outerHTML = ""
9871 </script>
9872 </span><-- end of class="gsh-src" -->
9873 <details>
9874 </details>
9875 </span>
9876 */
9877 <script id="GJLinkScript">
9878   function gjkey_hash(text){
9879     return strCRC32(text,text.length) % 0x10000;
9880   }
9881 }

```

```

9882 function gj_addlog(e,msg){
9883     now = (new Date().getTime() / 1000).toFixed(3);
9884     tstamp = '['+now+']';
9885     e.value = tstamp + msg;
9886     e.scrollTop = e.scrollHeight;
9887 }
9888 function gj_addlog_ci(msg){
9889     ws0_log.value += '(console.log) ' + msg + '\n';
9890 }
9891 var GJ_Channel = null;
9892 var GJ_Log = null;
9893 var gjx; // the global variable
9894 function GJ_Join(){
9895     target = gjx.join;
9896     if( target.value == 'Leave' ){
9897         GJ_Channel.close();
9898         GJ_Channel = null;
9899         target.value = 'Join';
9900     }
9901 }
9902 var ws0;
9903 var ws0_log;
9904 sav_console_log = console.error;
9905 console.error = gj_addlog_ci;
9906 ws0 = new WebSocket(gj_serv.innerHTML);
9907 console.error = sav_console_log
9908
9909 GJ_Channel = ws0;
9910 ws0_log = document.getElementById('ws0_log');
9911 GJ_Log = ws0_log;
9912
9913 now = (new Date().getTime() / 1000).toFixed(3);
9914 const wstats = ['CONNECTING','OPEN','CLOSING'];
9915 cst = wstats[ws0.readyState];
9916 gj_addlog(ws0_log, stat 'ws0.readyState'+cst+') : GJ Linked\n';
9917
9918 ws0.addEventListener('error', function(event){
9919     gj_addlog(ws0_log,'stat error : transport error?\n');
9920 });
9921 ws0.addEventListener('open', function(event){
9922     GJ_Channel = ws0;
9923     //console.log('#'+GJLinkView.id+' zIndex=' +GJLinkView.style.zIndex);
9924     date1 = new Date().getTime();
9925     date2 = (date1 / 1000).toFixed(3);
9926     seed = date1.toString(16);
9927
9928     // user name and key
9929     user = document.getElementById('gj_user').value;
9930     if( user.length == 0 ){
9931         gj_user.value = 'nemo';
9932         user = 'nemo';
9933     }
9934     key1 = document.getElementById('gj_ukey').bl_plainnext;
9935     ukey = gjkey_hash(seed+user+key1).toString(16);
9936
9937     // session name and key
9938     chan = document.getElementById('gj_chan').value;
9939     if( chan.length == 0 ){
9940         gj_chan.value = 'main';
9941         chan = 'main';
9942     }
9943     key2 = document.getElementById('gj_ckey').bl_plainnext;
9944     ckey = gjkey_hash(seed+chan+key2).toString(16);
9945
9946     msg = date2 + ' JOIN ' + user + '!' + chan + ' ' + ukey + ':' + ckey;
9947     gj_addlog(ws0_log,'send '+msg+'\n');
9948     ws0.send(msg);
9949
9950     target.value = 'Leave';
9951     //console.log('#'+date2+') #' +target.id+' '+target.value+'\n');
9952     //gj_addlog(ws0_log,'label '+target.value+'\n');
9953 );
9954 ws0.addEventListener('message', function(event){
9955     now = (new Date().getTime() / 1000).toFixed(3);
9956     msg = event.data;
9957     if( false ){
9958         gj_addlog(ws0_log,'recv '+msg+'\n');
9959     }
9960     argv = msg.split(' ')
9961     timestamp = argv[0];
9962     if( argv[1] == 'CAST' ){ // from reflector
9963         timestamp = argv[1];
9964         argv.shift(); // original time stamp
9965         ofrom = argv[0];
9966         argv.shift(); // original from
9967         argv.shift(); // command
9968         from = argv[0];
9969         argv.shift(); // from/to
9970         cmd1 = argv[0];
9971         argv.shift(); // xxxx command
9972         if( cmd1 == 'reload' ){
9973             location.reload()
9974         }
9975         gjcmd = argv[0];
9976         otstamp = '';
9977         if( gjcmd == 'CAST' ){ // from reflector
9978             otstamp = argv[0];
9979             argv.shift(); // original time stamp
9980             ofrom = argv[0];
9981             argv.shift(); // original from
9982             argv.shift(); // command
9983             from = argv[0];
9984             argv.shift(); // from/to
9985             cmd1 = argv[0];
9986             argv.shift(); // xxxx command
9987             if( false ){
9988                 gj_addlog(ws0_log,'--'
9989                     + ' tstamp=' + timestamp
9990                     + ' gjcmd=' + gjcmd
9991                     + ' from=' + from
9992                     + ' cmd1[' + cmd1 + ']'
9993                     + ' argv=' + argv);
9994             }
9995             if( cmd1 == 'auth' ){
9996                 // doing authorization required
9997             }
9998             if( cmd1 == 'echo' ){
9999                 cmd1 = (new Date()).getTime() / 1000).toFixed(3);
10000                 msg = now + '+' + RESP + 'join(' + );
10001                 gj_addlog(ws0_log,'send '+msg+'\n');
10002                 ws0.send(msg);
10003                 gj_addlog(ws0_log,'send '+msg+'\n');
10004             }
10005             if( cmd1 == 'eval' ){
10006                 argv.shift();
10007                 js = argv.join(' ');
10008                 re = eval(js); // <----- eval()
10009                 gj_addlog(ws0_log,'eval '+js+' = '+re+'\n');
10010                 now = (new Date().getTime() / 1000).toFixed(3);
10011                 msg = now + '+' + RESP + re;
10012                 ws0.send(msg);
10013                 gj_addlog(ws0_log,'send '+msg+'\n');
10014             }
10015         });
10016         ws0.addEventListener('close', function(event){
10017             if( GJ_Channel == null ){
10018                 gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
10019             }
10020             GJ_Channel.close();
10021             GJ_Channel = null;
10022             target.value = 'Join';
10023             target.value = 'Join';
10024             gj_addlog(ws0_log, stat error : close : GJ UnLinked unexpectedly\n');
10025         });
10026     }
10027     function GJ_BcastMessageUserPass(user,chan,msgbody){
10028         now = (new Date().getTime() / 1000).toFixed(3);
10029         msg = now + '+' + BCAST + 'user=' + chan + '+' + msgbody;
10030         if( false ){
10031             gj_addlog(GJ_Log,'send '+msg+'\n');
10032         }
10033         GJ_Channel.send(msg);
10034     }
10035     function GJ_BcastMessage(msgbody){
10036         if( GJ_Channel == null ){
10037             gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
10038             return;
10039         }
10040         //target = event.target;
10041         user = document.getElementById('gj_user').value;
10042         chan = document.getElementById('gj_chan').value;
10043         GJ_BcastMessageUserPass(user,chan,msgbody);

```

```

10044 }
10045     function GJ_SendMessageUserPass(user ,chan,msgbody){
10046         now = (new Date().getTime() / 1000).toFixed(3);
10047         msg = now + " ISAK "+user+'|'+chan+" "+msgbody;
10048         gj_adlog(GJ_Log,'send '+msg+'\n');
10049         GJ_Channel.send(msg);
10050     }
10051     function GJ_SendMessage(msgbody){
10052         if( GJ_Channel == null ){
10053             gj_adlog(wso_Log,'stat error : send : GJ not Linked\n');
10054             return;
10055         }
10056         //target = event.target;
10057         user = document.getElementById('GJ_user').value;
10058         chan = document.getElementById('GJ_chan').value;
10059         GJ_SendMessageUserPass(user,chan,msgbody);
10060     }
10061     function GJ_Send(){
10062         msgbody = gj_sendText.value;
10063         GJ_SendMessage(msgbody);
10064     }
10065 </script>
10066
10067 <!-- ----- GJLINK ----- -->
10068 <!--
10069     - User can subscribe to a channel
10070     - A channel will be broadcasted
10071     - A channel can be a pattern (regular expression)
10072     - User is like From(me) and channel is like To: or Recipient:
10073     - like VIABUS
10074     - watch message with SENDME, WATCH, CATCH, HEAR, or so
10075     - routing with path expression or name pattern (with routing with DNS like system)
10076-->
10077 */
10078
10079 //<span id="GJLinkGolang">
10080 //<details id="GshWebsocket"><summary>Golang / JavaScript Link</summary>
10081 //<span class="close">Close</span></details><!-- { -->
10082 // 2020-0920 created
10083 //<a href="https://pkgs.go.dev/golang.org/x/net/websocket">WS</a>
10084 //<a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
10085 // INSTALL go get golang.org/x/net/websocket
10086 //<pre>go get golang.org/x/net/websocket
10087 //<pre>go install ./net/websocket
10088 // import "golang.org/x/net/websocket"
10089 const gshws_origin = "http://localhost:9999"
10090 const gshws_server = "localhost:9999"
10091 const gshws_port = "9999"
10092 const gshws_path = "/gjlink"
10093 const gshws_url = "ws://" + gshws_origin + gshws_path
10094 const GSHWS_MSGSIZE = 61024
10094 func fmtstring(fmts string, params ...interface{}) (string){
10095     return fmt.Sprintf(fmts,params...)
10096 }
10097 func GSHWS_MARK(what string)(string{
10098     now := time.Now()
10099     us := fmt.Sprintf("%06d",now.Nanosecond() / 1000)
10100    mark := ""
10101    if( !AtConsoleLineTop ){
10102        mark += "\n"
10103        AtConsoleLineTop = true
10104    }
10105    mark += "["+now.Format(time.Stamp)+"."+us+"] -GJ- " + what + " "
10106    return mark
10107 }
10108 func gohk(what string,err error){
10109     if( err != nil ){
10110         panic(GSHWS_MARK(what)+err.Error())
10111     }
10112 }
10113 func glog(what string, fmts string, params ...interface{}{
10114     fmt.Println(GSHWS_MARK(what))
10115     fmt.Printf(fmts+"\n",params...)
10116 }
10117
10118 var WSV = []*websocket.Conn
10119 func jsend(argv []string{
10120     if len(argv) <= 1 {
10121         fmt.Printf("-ij %v [-m] command arguments\n",argv[0])
10122         return
10123     }
10124     argv = argv[1:]
10125     if( len(WSV) == 0 ){
10126         fmt.Println("-Ej-- No link now\n")
10127         return
10128     }
10129     if( 1 < len(WSV) ){
10130         fmt.Println("-ij-- multiple links (%v)\n",len(WSV))
10131     }
10132
10133     multicast := false // should be filtered with regexp
10134     if( 0 < len(argv) && argv[0] == "-m" ){
10135         multicast = true
10136         argv = argv[1];
10137     }
10138     args := strings.Join(argv, " ")
10139
10140     now := time.Now()
10141     msec := now.UnixNano() / 1000000;
10142     timestamp := fmt.Sprintf("%.3f",float64(msec)/1000.0)
10143     msg := fmt.Sprintf("%v SEND gshell[* %v",timestamp,args)
10144
10145     if( multicast ){
10146         for i,ws := range WSV {
10147             wn,werr := ws.Write([]byte(msg))
10148             if( werr != nil ){
10149                 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10150             }
10151             glog("SQ",fmtstring("(%v) %v",wn,msg))
10152         }
10153     } else{
10154         i := 0
10155         ws := WSV[i]
10156         wn,werr := ws.Write([]byte(msg))
10157         if( werr != nil ){
10158             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10159         }
10160         glog("SQ",fmtstring("(%v) %v",wn,msg))
10161     }
10162 }
10163 func ws_broadcast(msg string) (wn int,werr error){
10164     for i,ws := range WSV {
10165         wn,werr = ws.Write([]byte(msg))
10166         if( werr != nil ){
10167             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10168         }
10169         glog("SQ",fmtstring("(%v) %v",wn,msg))
10170     }
10171     return wn,werr;
10172 }
10173 func servilws *websocket.Conn {
10174     WSV = append(WSV,ws)
10175     //fmt.Println("n")
10176     glog("CO","accepted connections(%v)",len(WSV))
10177     //remoteAddr := ws.RemoteAddr()
10178     //fmt.Println("-- accepted %v",remoteAddr)
10179     //fmt.Println("-- accepted %v",ws.Config())
10180     //fmt.Println("-- accepted %v",ws.Config().Header)
10181     //fmt.Println("-- accepted %v // %v",ws.servi)
10182
10183     var reqb = make([]byte,GSHWS_MSGSIZE)
10184     for {
10185         rn, rerr := ws.Read(reqb)
10186         if( rerr != nil || rn < 0 ){
10187             glog("SQ",fmtstring("(%v,%v)",rn,rerr))
10188             break;
10189         }
10190         req := string(reqb[0:rn])
10191         glog("SQ",fmtstring("(%v) %v",rn,req))
10192
10193         argv := strings.Split(req, " ");
10194         argv = argv[1];
10195         if( 0 < len(argv) ){
10196             if( argv[0] == "RESP" ){
10197                 // should forward to the destination
10198                 continue;
10199             }
10200         }
10201         now := time.Now()
10202         msec := now.UnixNano() / 1000000;
10203         timestamp := fmt.Sprintf("%.3f",float64(msec)/1000.0)
10204         res := fmtstring("%v "+CAST+" %v",timestamp,req)
10205

```

```

10206     wn := 0;
10207     werr := error(nil);
10208     if( 0 < len(margv) && margv[0] == "BCAST" ){
10209         wn, werr = ws.broadcast(res);
10210     }else{
10211         wn, werr = ws.Write([]byte(res))
10212     }
10213     gchk("SE",werr)
10214     glog("SR",fmtstring("(#v) tv",wn,string(res)))
10215 }
10216 glog("SF","Ws response finish")
10217
10218 wsv := []websocket.Conn{}
10219 wsx := []
10220 for i,v := range WSV {
10221     if(v != ws){
10222         wsx = i
10223         wsv = append(wsv,v)
10224     }
10225 }
10226 WSV = wsx
10227 //glog("CO","closed #v",ws)
10228 glog("CO","closed connection [#v/#v]",wsx+1,len(WSV)+1)
10229 ws.Close()
10230 }
10231 url := [:scheme[://host[:port]]/:path]
10232 func decomp_URL(url string){
10233 }
10234 func full_wsURL(){
10235 }
10236 func gj_server(argv []string) {
10237     gjserv := gshws_url
10238     gjport := gshws_server
10239     gjpath := gshws_path
10240     gjscheme := "ws"
10241
10242 //cmd := argv[0]
10243     argv = argv[1:]
10244     if( 1 <= len(argv) ){
10245         serv := argv[0]
10246         if( 0 < strings.Index(serv,"/") ){
10247             schemev := strings.Split(serv,"/")
10248             gjscheme = schemev[0]
10249             serv = schemev[1]
10250         }
10251         if( 0 < strings.Index(serv,"/") ){
10252             pathv := strings.Split(serv,"/")
10253             serv = pathv[0]
10254             gjpath = pathv[1]
10255         }
10256         servv := strings.Split(serv,":")
10257         host := "localhost"
10258         port := 9999
10259         if( servv[0] != "" ){
10260             host = servv[0]
10261         }
10262         if( len(servv) == 2 ){
10263             fmt.Sscanf(servv[1],"%d",&port)
10264         }
10265         //glog("LC","hostport=%v (%v : %v)",servv.host,port)
10266         gjport = fmt.Sprintf("%v:%v",host,port)
10267         gjserv = gjscheme + "//" + gjport + "/" + gjpath
10268     }
10269     glog("LS",fmtstring("listening at %v",gjserv))
10270     http.Handle(fmt.Sprintf("/%s",gjserv),websocket.Handler(serv))
10271     err := errnil()
10272     if( gjscheme == "ws" ){
10273         // https://golang.org/pkg/net/http/#ListenAndServeTLS
10274         //err = http.ListenAndServeTLS(gjport,nil)
10275     }else{
10276         err = http.ListenAndServe(gjport,nil)
10277     }
10278     gchk("LB",err)
10279 }
10280
10281 func gj_client(argv []string) {
10282     glog("CS",fmtstring("connecting to %v",gshws_url))
10283     ws, err := websocket.Dial(gshws_url,"",gshws_origin)
10284     gchk("C",err)
10285
10286     var resb = make([]byte, GSHWS_MSGSIZE)
10287     for qi := 0; qi < 3; qi++ {
10288         req := fmtstring("Hello, GShell! (%v)",qi)
10289         ws.Write([]byte(req))
10290         glog("QW",fmtstring("(#v) tv",wn,req))
10291         gchk("QE",werr)
10292         rn, rerr := ws.Read(resb)
10293         gchk("RE",rerr)
10294         glog("RM",fmtstring("(#v) tv",rn,string(resb)))
10295     }
10296     glog("CF","Ws request finish")
10297 }
10298 //</span><-- end of class="gsh-src" >-->
10299 //</details></span>
10300
10301 /*
10302 <details id="GJLink_Section"><summary>GJ Link</summary>
10303 <span id="GJLinkView" class="GJLinkView">
10304 <p>
10305 <note class="GJNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
10306 </p>
10307
10308 <span id="GJLink_1">
10309 <div id="GJLink_ServerSet"></div>
10310
10311 <div>
10312 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
10313 <span id="GJLink_Account"></span>
10314 </div>
10315
10316 <div>
10317 <input id="gj_sendbutton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
10318 <span id="GJLink_SendArea"></span>
10319 </div>
10320
10321 <div id="ws0_log_container"></div>
10322 </span>
10323 </span>
10324
10325 <script>
10326 function setupGJLinkArea(){
10327     GJLink_ServerSet.innerHTML = '<'+span id="gj_serv_label" +
10328         + ' class="textLabel textLabel">Server: <'+span>
10329         + '<'+span id="gj_serv" class="textField textURL" contenteditable=<'+'>span';
10330
10331     GJLink_Account.innerHTML = '<'+textarea id="gj_user" class="textField"><'+/textarea>
10332         + '<'+textarea id="gj_ukey" class="textField"><'+/textarea>
10333         + '<'+textarea id="gj_chan" class="textField"><'+/textarea>
10334         + '<'+textarea id="gj_key" class="textField"><'+/textarea>;
10335
10336     GJLink_SendArea.innerHTML =
10337         + '<'+textarea id="gj_sendText" class="textField MsgText" cols=60 rows=2><'+/textarea>;
10338
10339     ws0_log_container.innerHTML = '<'+textarea id="ws0_log" class="ws0_log" +
10340         + 'cols=100 rows=10 spellcheck="false"><'+/textarea>';
10341 }
10342 function clearGJLinkArea(){
10343     GJLink_ServerSet.innerHTML = "";
10344     GJLink_Account.innerHTML = "";
10345     GJLink_SendArea.innerHTML = "";
10346     ws0_log_container.innerHTML = "";
10347 }
10348 </script>
10349
10350 <script>
10351 function SetupGJLink(){
10352     SetupGJLinkArea();
10353     SetupVisibleText(GJLink_1,gj_serv,'GJLinkSv');
10354     SetupVisibleText(GJLink_1,gj_user,'UserName');
10355     SetupBlinderText(GJLink_1,gj_ukey,'UserKey');
10356     SetupVisibleText(GJLink_1,gj_chan,'ChannelName');
10357     SetupBlinderText(GJLink_1,gj_ckey,'ChannelKey');
10358     SetupVisibleText(GJLink_1,gj_sendtext,'Message');
10359     gj_serv.innerHTML = "ws://localhost:9999/gjlink1";
10360 }
10361 function GJLink_init(){
10362     SetupGJLink();
10363 }
10364 function iselem(eid){
10365     return document.getElementById(eid);
10366 }
10367 function DestroyGJLink1(){

```

```

10368 clearGJLinkKrea();
10369 if( !iselem('gj_user') ){
10370     return;
10371 }
10372 if( gj_serv_label.parentNode != gj_user ){
10373     return;
10374 }
10375 if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
10376 if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
10377 if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
10378 if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
10379 if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
10380 if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
10381 if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
10382 if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
10383 }
10384 DestroyGJLink = DestroyGJLink1;
10385 </script>
10386 </details>
10387 </div>
10388 /*
10389 <style>
10390 .GxDigest {
10391     display:none;
10392 }
10393 </style>
10394 <script id="HtmlCodeview-script">
10395     function showNodeAsHtmlSource(xtxa,code,prefix,postfix,sign){
10396         txa = document.createElement('textarea');
10397         txa.id = 'otxa_id';
10398         txa.setAttribute('class', 'HtmlCodeviewText');
10399         otxa.parentNode.replaceChild(txa,otxa);
10400         otxa.setAttribute('spellcheck','false');
10401         /txa.value = code.innerHTML;
10402         /txa.innerHTML = code.innerHTML + postfix;
10403         if( sign ){
10404             text = txa.value;
10405             tlen = txa.value.length;
10406             digest = strCRC32(text,tlen) + ' ' + tlen
10407             + code.id + (' ' + DateShort()) + ')';
10408             //alert('digest: '+digest);
10409             console.log('digest: '+digest);
10410             txa.innerHTML += '<+'+span class="GxDigest">' + digest + '</' + span>\n';
10411         }
10412         txa.style.display = "block";
10413         txa.style.width = "100%";
10414         txa.style.height = "300px";
10415     }
10416     function showNodeAsHtmlCode(xtxa,code,prefix,postfix,sign){
10417         if( event.target.value == 'ShowCode' ){
10418             showNodeAsHtmlSourceX(otxa,code,prefix,postfix,sign);
10419             event.target.value = 'HideCode';
10420         } else{
10421             otxa.style.display = "none";
10422             event.target.value = 'ShowCode';
10423         }
10424     }
10425     function showNodeAsHtmlSource(otxa,code){
10426         showNodeAsHtmlSourceX(otxa,code,'','');
10427     }
10428     function showNodeAsHtmlCode(otxa,code){
10429         if( event.target.value == 'ShowCode' ){
10430             showNodeAsHtmlSource(otxa,code);
10431             event.target.value = 'HideCode';
10432         } else{
10433             otxa.style.display = "none";
10434             event.target.value = 'ShowCode';
10435         }
10436     }
10437 </script>
10438 <style id="HtmlCodeview-style">
10439     .HtmlCodeviewText {
10440         font-size:10pt;
10441         font-family:Courier New;
10442         white-space:pre;
10443     }
10444     .HtmlCodeviewButton {
10445         padding:2px !important;
10446         line-height:1.1 !important;
10447         border:2px inset #bbb !important;
10448         font-size:11pt !important;
10449         font-weight:bold !important;
10450         font-family:Georgia !important;
10451         border-radius:3px !important;
10452         border:#ddd; background-color:#228 !important;
10453         color:#fff;
10454     }
10455 </style>
10456 /*
10457 <details><summary>Live HTML Snapshot</summary>
10458 <span id="LiveHTML">
10459     <span>Live HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
10460     <div class="GahMenu">
10461         <span class="GahMenu1" onclick="html_edit()>">Edit</span>
10462         <span class="GahMenu1" onclick="html_save()>">Save</span>
10463         <span class="GahMenu1" onclick="html_load()>">Load</span>
10464         <span class="GahMenu1" onclick="html_ver0()>">Vers</span>
10465     </div>
10466     <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLcode()">
10467     <span id="LiveHTML_Codeview"></span>
10468 </div>
10469 <script id="LiveHTML_Script">
10470     function showLiveHTMLcode(){
10471         showHtmlCode(LiveHTML_Codeview,LiveHTML);
10472     }
10473     var _editable = false;
10474     var savSuppressGJShell = false;
10475     function ToggleEditMode(){
10476         if( _editable ){
10477             savSuppressGJShell = SuppressGJShell;
10478             SuppressGJShell = true;
10479             gsh.setAttribute('contenteditable','true');
10480             GahMenuEdit.getAttribute('innerHTML') = 'Lock';
10481             GahMenuEdit.style.color = 'rgba(255,0,0,1)';
10482             GahMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10483         } else{
10484             SuppressGJShell = savSuppressGJShell;
10485             gsh.setAttribute('contenteditable','false');
10486             GahMenuEdit.getAttribute('innerHTML') = 'Edit';
10487             GahMenuEdit.style.color = 'rgba(16,160,16,1)';
10488             GahMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10489         }
10490     }
10491     function html_edit(){
10492         ToggleEditMode();
10493     }
10494     function html_save(){
10495         // Live HTML (DOM) Snapshot onto browser's localStorage
10496         // 2020-0923 SatoxITS
10497         var htroot = gsh // - Element-ID, should be selectable
10498         const snappeditHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStorage
10499         // -- should be a (map) of URL
10500         // -- should be with CSSOM as inline script
10501         const htVersionTag = 'VersionTag'; // VersionTag Element-ID in the HTML (in DOM)
10502         function showVersion(note,w,v,u,t){
10503             w.alert(note);
10504             v += 'w=' + w + '\n';
10505             v += 'v=' + v + '\n';
10506             v += 'u=' + u + '\n';
10507             v += 't=' + t + ' (' + DateLong0(t*1000) + ')';
10508         }
10509         function html_save(){
10510             var ht = document.HTM;
10511             t = new Date().getTime() / 1000;
10512             v = '<' + span id="htVersionTag"' data-url="'+u+'" data-time="'+t+'>';
10513             v += '<+' + span>\n';
10514             h += v + htroot.outerHTML;
10515             localStorage.setItem(snappeditHTML,h);
10516             showVersion("Saved",window,v,u,t);
10517         }
10518         function html_load(){
10519             h = localStorage.getItem(snappeditHTML);
10520             if( h == null ){
10521                 alert('No snapshot taken yet');
10522                 return;
10523             }
10524             w = window.open('','','');
10525             w = w.document;
10526         }
10527     }
10528 
```

```

10530 d.write(h);
10531 w.focus();
10532 html_ver1("Loaded",w,d);
10533 }
10534 function html_ver1(note,w,d){
10535 if( (v = d.getElementById(htVersionTag)) != null ){
10536 h = v.outterHTML;
10537 u = v.getAttribute('data-url');
10538 t = v.getAttribute('data-time');
10539 }else{
10540 h = 'No version info. in the page';
10541 u = '';
10542 t = 0;
10543 }
10544 showVersion(note,w,v,u,t);
10545 }
10546 function html_ver0(){
10547 html_ver1("Version",window,document);
10548 }
10549 <script>
10550 </span>LiveHTML } -->
10551 </span>
10552 </details>
10553</
10554 <details><summary>Event sharing</summary>
10555 <span id="EventSharingCodeSpan">
10556 <!-- ----- Event sharing // 2020-0925 SatoxITS { -->
10557 <div id="lifetemplate" class="lifet" hidden=">
10558 <style> .lifetbody{ color:#22;font-family:Georgia;font-size:10pt; } </style>
10559 <span id="framebody" class="lifetbody" onclick="frameClick()"><script>
10560 function docadd(txt){
10561 document.body.append(txt);
10562 window.scrollTo(0,100000);
10563 }
10564 function frameClick(){
10565 xy = "(x*+event.x + 'y'+event.y)*";
10566 //docadd('Got Click on #' + event.target.id + ' ' + xy + '\n');
10567 docadd('Got Click on #' + Fid.value + ' ' + xy + '\n');
10568 window.scrollTo(0,100000);
10569 window.parent.postMessage('OnClick: ' + xy, '*');
10570 }
10571 function frameMousemove(){
10572 if( false ){
10573 document.body.append('Mousemove on #' + event.target.id + ' ' +
10574 + 'x' + event.x + 'y'+event.y + '\n');
10575 peerWin = window.frames[frame];
10576 document.body.append('Send to peer #' + peerWin + ' ' + '\n');
10577 window.scrollTo(0,100000);
10578 peerWin.postMessage('Hi','');
10579 }
10580 }
10581 function framekeydown(){
10582 msg = 'Got Keydown: #' + Fid.value + ', (' + event.code + ')';
10583 docadd(msg + '\n');
10584 window.parent.postMessage(msg, '*');
10585 }
10586 function frameonmessage(){
10587 docadd('Message ' + event.data + '\n');
10588 window.scrollTo(0,100000);
10589 }
10590 if( document.getElementById('Fid') ){
10591 framebody.id = Fid.value;
10592 h = '<'+style+'<';
10593 h += 'font-size:10pt;white-space:pre-wrap;';
10594 h += 'font-family:Courier New;';
10595 h += 'color:' + Fid.value;
10596 h += 'background-color:' + Fid.value + '\n';
10597 document.write(h);
10598 window.addEventListener('click',frameClick);
10599 window.addEventListener('keydown',framekeydown);
10600 window.addEventListener('message',frameonmessage);
10601 window.addEventListener('mousemove',framemousemove);
10602 window.parent.postMessage('Hi parent, I am ' + Fid.value, '*');
10603 }
10604 </script></span></div>
10610 <style>.iframeTest{margin:3px;font-size:10pt;"></style>
10611 <div id="iframeTest" style="border:1px solid black; width:370px; height:120px;">
10612 <h2>Inter-window communication</h2>
10613 <br>
10614 frame0 >> frame1 and frame2<br>
10615 frame1 >> frame0 and frame2<br>
10616 frame2 >> frame0 and frame1<br>
10617 </note>
10618 <div id="iframe-test">
10619 <div id="iframehost" style="border:1px solid black; font-family:Courier New;font-size:10pt;"></pre>
10620 <frame id="frame0" title="frame0" class="iframeTest" style=""></frame>
10621 <frame id="frame1" title="frame1" class="iframeTest"></frame>
10622 <frame id="frame2" title="frame2" class="iframeTest"></frame>
10623 </div>
10624 <script id="if0-test-script">
10625 function InterFrameComm_init(){
10626 setupFrames0();
10627 setupFrames12();
10628 }
10629 function setFrameSrcDoc(dst,src){
10630 if( true ){
10631 dst.contentWindow.document.write(src);
10632 // this makes browser waste close, and crash if accumulated !?
10633 // so it should be closed after write
10634 dst.contentWindow.document.close();
10635 }
10636 else{
10637 // to be erased before source dump
10638 // but should be set for live snapshot
10639 dst.srccode = src;
10640 }
10641 }
10642 function setupFrames(){
10643 iBody = iframe0.contentWindow.document.body;
10644 iframe0.style.width = "755px";
10645 //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
10646 window.addEventListener('message',messageFromChild);
10647 if0 = '';
10648 if0 += '';
10649 if0 += '<'+pre style="font-family:Courier New;">';
10650 if0 += '<input id="Fid" value="iframe0">';
10651 if0 += iftestTemplate.innerHTML;
10652 setFrameSrcDoc(iframe0,if0);
10653 }
10654 function clickOnChild(){
10655 console.log('clickon #' + this.id);
10656 }
10657 function moveOnChild(){
10658 console.log('moveon #' + this.id);
10659 }
10660 iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
10661 iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
10662 }
10663 function setupFrames12(){
10664 if1 = '<input id="Fid" value="iframe1">';
10665 if1 += iftestTemplate.innerHTML;
10666 setFrameSrcDoc(iframe1,if1);
10667 //iframe1.name = "iframe1"; // this seems break contentWindow
10668 if2 = '<input id="Fid" value="iframe2">';
10669 if2 += iftestTemplate.innerHTML;
10670 setFrameSrcDoc(iframe2,if2);
10671 }
10672 iframe1.addEventListener('message',messageFromChild);
10673 //iframe1.addEventListener('mouseover',moveOnChild);
10674 iframe1.addEventListener('message',messageFromChild);
10675 //iframe2.addEventListener('mouseover',moveOnChild);
10676 //iframe2.addEventListener('message',messageFromChild);
10677 iframe1.contentWindow.postMessage('[parent0] Hi iframe1 -- from parent.','*');
10678 //iframe1.contentWindow.postMessage('[parent0] Your peer is '+iframe2.contentWindow,'*');
10679 iframe2.contentWindow.postMessage('[parent0] Hi iframe2 -- from parent.','*');
10680 //iframe2.contentWindow.postMessage('[parent0] Your peer is '+iframe1.contentWindow,'*');
10681 }
10682 function messageFromChild(){
10683 from = null;
10684 forw = null;
10685 if( event.source == iframe0.contentWindow ){
10686 from = '[iframe0]';
10687 forw = 'iframe12';
10688 }
10689 if( event.source == iframe1.contentWindow ){
10690 from = '[iframe1]';
10691 forw = 'iframe2';
10692 }
10693 }
10694 }
10695 }
10696 }
10697 }
10698 }
10699 }
10700 }
10701 }
10702 }
10703 }
10704 }
10705 }
10706 }
10707 }
10708 }
10709 }
10710 }
10711 }
10712 }
10713 }
10714 }
10715 }
10716 }
10717 }
10718 }
10719 }
10720 }
10721 }
10722 }
10723 }
10724 }
10725 }
10726 }
10727 }
10728 }
10729 }
10730 }
10731 }
10732 }
10733 }
10734 }
10735 }
10736 }
10737 }
10738 }
10739 }
10740 }
10741 }
10742 }
10743 }
10744 }
10745 }
10746 }
10747 }
10748 }
10749 }
10750 }
10751 }
10752 }
10753 }
10754 }
10755 }
10756 }
10757 }
10758 }
10759 }
10760 }
10761 }
10762 }
10763 }
10764 }
10765 }
10766 }
10767 }
10768 }
10769 }
10770 }
10771 }
10772 }
10773 }
10774 }
10775 }
10776 }
10777 }
10778 }
10779 }
10780 }
10781 }
10782 }
10783 }
10784 }
10785 }
10786 }
10787 }
10788 }
10789 }
10790 }
10791 }
10792 }
10793 }
10794 }
10795 }
10796 }
10797 }
10798 }
10799 }
10800 }
10801 }
10802 }
10803 }
10804 }
10805 }
10806 }
10807 }
10808 }
10809 }
10810 }
10811 }
10812 }
10813 }
10814 }
10815 }
10816 }
10817 }
10818 }
10819 }
10820 }
10821 }
10822 }
10823 }
10824 }
10825 }
10826 }
10827 }
10828 }
10829 }
10830 }
10831 }
10832 }
10833 }
10834 }
10835 }
10836 }
10837 }
10838 }
10839 }
10840 }
10841 }
10842 }
10843 }
10844 }
10845 }
10846 }
10847 }
10848 }
10849 }
10850 }
10851 }
10852 }
10853 }
10854 }
10855 }
10856 }
10857 }
10858 }
10859 }
10860 }
10861 }
10862 }
10863 }
10864 }
10865 }
10866 }
10867 }
10868 }
10869 }
10870 }
10871 }
10872 }
10873 }
10874 }
10875 }
10876 }
10877 }
10878 }
10879 }
10880 }
10881 }
10882 }
10883 }
10884 }
10885 }
10886 }
10887 }
10888 }
10889 }
10890 }
10891 }
10892 }
10893 }
10894 }
10895 }
10896 }
10897 }
10898 }
10899 }
10900 }
10901 }
10902 }
10903 }
10904 }
10905 }
10906 }
10907 }
10908 }
10909 }
10910 }
10911 }
10912 }
10913 }
10914 }
10915 }
10916 }
10917 }
10918 }
10919 }
10920 }
10921 }
10922 }
10923 }
10924 }
10925 }
10926 }
10927 }
10928 }
10929 }
10930 }
10931 }
10932 }
10933 }
10934 }
10935 }
10936 }
10937 }
10938 }
10939 }
10940 }
10941 }
10942 }
10943 }
10944 }
10945 }
10946 }
10947 }
10948 }
10949 }
10950 }
10951 }
10952 }
10953 }
10954 }
10955 }
10956 }
10957 }
10958 }
10959 }
10960 }
10961 }
10962 }
10963 }
10964 }
10965 }
10966 }
10967 }
10968 }
10969 }
10970 }
10971 }
10972 }
10973 }
10974 }
10975 }
10976 }
10977 }
10978 }
10979 }
10980 }
10981 }
10982 }
10983 }
10984 }
10985 }
10986 }
10987 }
10988 }
10989 }
10990 }
10991 }
10992 }
10993 }
10994 }
10995 }
10996 }
10997 }
10998 }
10999 }
109999 }
```

```

10692     else
10693     if( event.source == iframe2.contentWindow ){
10694         from = '[iframe2]';
10695         forw = 'iframe1';
10696     else
10697     {
10698         iframeHost.innerHTML += 'Message [unknown] '
10699         + ' orig=' + event.origin
10700         + ' data=' + event.data
10701         //+ ' from=' + event.source
10702         ;
10703     }
10704     msglog1 = from + event.data + ' -- '
10705     + ' from=' + event.source
10706     + ' orig=' + event.origin
10707     + ' name=' + event.source.name
10708     //+ ' ports' + event.ports
10709     //+ ' evid=' + event.lastEventId
10710     + '\n'
10711     ;
10712     if(true){
10713         if(forw == 'iframe1' || forw == 'iframe12'){
10714             iframe1.contentWindow.postMessage(from+event.data);
10715         }
10716         if( forw == 'iframe2' || forw == 'iframe12' ){
10717             iframe2.contentWindow.postMessage(from+event.data);
10718         }
10719     }
10720     txtadd0(msglog1);
10721 }
10722 function txtadd0(txt){
10723     iframe0.contentWindow.document.body.append(txt);
10724     iframe0.contentWindow.scrollTo(0,10000);
10725 }
10726 }
10727 function es_ShowSelf(){
10728     iframe1.setAttribute('src',document.URL);
10729     iframe2.setAttribute('src',document.URL);
10730 }
10731 </script>
10732
10733 <input class="HtmlCodeViewbutton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
10734 <input class="HtmlCodeViewbutton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
10735 <span id="EventSharingDevview"></span>
10736 <script id="EventSharingScript">
10737 function es_showHtmlCode(){
10738     showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
10739 }
10740 DestroyEventSharingCodeview = function(){
10741     /EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
10742     EventSharingCodeview.innerHTML = "";
10743     iframe0.style = "";
10744     //iframe0.srccode = "erased";
10745     //iframe1.srccode = "erased";
10746     //iframe2.srccode = "erased";
10747 }
10748 </script>
10749 <!-- EventSharing -->
10750 </span>
10751 </details>
10752 </div>
10753
10754 /*
10755 <!-- ----- "GShell Inside" Notifikasiacion ( -->
10756 <script id="script-gshell-inside">
10757 var notices = 0;
10758 function noticeGShellInside(){
10759     ver = '';
10760     if( ver = document.getElementById('GshVersion') ) {
10761         ver = ver.innerHTML;
10762     }
10763     console.log('GJShell Inside (*~)/*'+ver);
10764     notices += 1;
10765     if( 2 <= notices ) {
10766         document.removeEventListener('mousemove',noticeGShellInside);
10767     }
10768 }
10769 document.addEventListener('mousemove',noticeGShellInside);
10770 noticeGShellInside();
10771
10772 const FooterName = 'GshFooter';
10773 function DestroyFooter(){
10774     if( (footer = document.getElementById(FooterName)) != null ){
10775         footer.parentNode.removeChild(footer);
10776         empty = document.createElement('div');
10777         empty.id = 'GshFooter0';
10778         footer.parentNode.replaceChild(empty,footer);
10779     }
10780 }
10781 function showFooter(){
10782     footer = document.createElement('div');
10783     footer.id = FooterName;
10784     footer.style.backgroundImage = "url("+ITSmoreQR+"")";
10785     /GshFooter0.parentNode.appendChild(footer);
10786     GshFooter0.parentNode.replaceChild(footer,GshFooter0);
10787 }
10788 </script>
10789 <!-- -->
10790
10791 <!-- border:20px inset #888;
10792 -->
10793
10794
10795 //<span id="VirtualDesktopCodeSpan">
10796 /*
10797 <details id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
10798 <!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
10799 <style>
10800 .Virtualspace {
10801     z-index:0;
10802     width:1280px !important; xheight:720px !important;
10803     width:5120px !important; height:2880px;
10804     background-color:#fff;
10805     xbackground-color:rgba(32,32,160,0.8);
10806     xbackground-image:url("WD-WallPaper03.png");
10807     xbackground-size:100% 100%;
10808     color:#22a; font-family:Georgia;font-size:10pt;
10809     xoverflow:scroll;
10810 },
10811 .Virtualgrid {
10812     z-index:0;
10813     position:absolute;
10814     width:800px; height:500px;
10815     border:1px solid #fff;
10816     color:rgba(192,255,192,0.8);
10817     font-family:Georgia, Courier New;
10818     text-align:right;
10819     vertical-align:middle;
10820     font-size:200px;
10821     text-shadow:4px 4px #ccf;
10822 },
10823 .WGridscroll {
10824     z-index:100000;
10825     background-color:rgba(200,200,200,0.1);
10826 },
10827 .Virtualdesktop {
10828     z-index:0;
10829     position:relative;
10830     resize:both !important;
10831     overflow:scroll;
10832     display:block;
10833     min-width:120px !important; min-height:60px !important;
10834     width:800px;
10835     height:500px;
10836     border:10px inset #228;
10837     border-width:30px; border-radius:20px;
10838     background-image:url("WD-WallPaper03.png");
10839     background-size:100% 100%;
10840     color:#22a; font-family:Georgia;font-size:10pt;
10841 },
10842 .comment {
10843     // overflow=scroll seems to bound childrens' view in the element span
10844     // specifying overflow seems fix the position of the element
10845 },
10846 .VirtualbrowserSpan {
10847     z-index:10;
10848     xxxorder:0.5px dashed #fff !important;
10849     border-color:rgba(255,255,255,0.5) !important;
10850     position: relative;
10851     left:100px;
10852     top:100px;
10853     display: block;

```

```

10854 resize:both !important;
10855 width:540px;
10856 height:30px;
10857 min-width:40px !important; min-height:20px !important;
10858 max-width:5120px !important; max-height:2880px !important;
10859 background-color:rgba(255,200,255,0.1);
10860 overflow:scroll;
10861 }
10862 .xVirtualBrowserLocationBar:active {
10863 color:#f00;
10864 background-color:colorrgba(255,128,128,0.2);
10865 }
10866 .xVirtualBrowserLocationBar:active {
10867 color:#f00;
10868 background-color:rgba(128,255,128,0.2);
10869 }
10870 a.VirtualBrowserLocation {
10871 color:#ccc !important;
10872 text-decoration:none !important;
10873 }
10874 a.VirtualBrowserLocation:hover {
10875 color:#fff !important;
10876 text-decoration:underline;
10877 }
10878 .VirtualBrowserlocationBar {
10879 position:fixed;
10880 z-index:200000;
10881 display:block;
10882 width:400px;
10883 height:20px;
10884 padding-left:1px;
10885 line-height:1.1;
10886 vertical-align:middle;
10887 font-size:14px;
10888 color:#fff;
10889 background-color:colorrgba(128,128,128,0.2);
10890 font-family:Georgia;
10891 }
10892 .VirtualBrowserCommandBar {
10893 position:absolute;
10894 z-index:200000;
10895 xxdisplay:inline;
10896 display:block;
10897 width:60px;
10898 height:20px;
10899 line-height:1.1;
10900 vertical-align:middle;
10901 font-size:14px;
10902 color:#f00;
10903 background-color:colorrgba(128,128,128,0.1);
10904 font-family:Georgia;
10905 text-align:left;
10906 left:404px;
10907 }
10908 .VirtualBrowserFrame {
10909 xposition:relative;
10910 position:absolute;
10911 xxdisplay:inline;
10912 display:block;
10913 z-index:100;
10914 resize:both !important;
10915 width:480px; height:240px;
10916 min-width:60px; min-height:30px;
10917 max-width:5120px; max-height:2880px;
10918 border-radius:6px;
10919 border-top:2px solid;
10920 border-left:2px solid;
10921 border-right:4px solid;
10922 border-bottom:1px solid;
10923 }
10924 .WinCaption {
10925 width:16px;
10926 height:16px;
10927 margin:0px;
10928 margin-right:3px;
10929 vertical-align:middle;
10930 background-color:colorrgba(255,255,255,1.0);
10931 }
10932 .VirtualdesktopMenuBar {
10933 xposition:absolute;
10934 color:#fff;
10935 font-size:7pt;
10936 text-align:right;
10937 padding-right:4px;
10938 background-color:colorrgba(128,128,128,0.7);
10939 }
10940 .VirtualdesktopCalender {
10941 color:#fff;
10942 font-size:9pt;
10943 text-align:right;
10944 padding-right:4px;
10945 xbackground-color:colorrgba(255,255,255,0.2);
10946 }
10947 xxxxinput { width:260px !important; line-height:1.1 !important; margin:0px;
10948 display:inline !important; font-size:10pt !important; padding:1px !important;
10949 }
10950 .WD_Config {
10951 display:inline !important;
10952 padding:2px !important;
10953 font-size:10px !important;
10954 width:60px !important;
10955 height:12pt !important;
10956 line-height:1.0pt !important;
10957 height:15px !important;
10958 }
10959 .WD_Button {
10960 display:inline !important;
10961 padding:2px !important;
10962 color:#fff !important;
10963 background-color:#228B22 !important;
10964 font-size:10pt !important;
10965 width:60pt !important;
10966 height:12pt !important;
10967 line-height:1.0pt !important;
10968 height:16pt !important;
10969 border:2px inset #44a !important;
10970 }
10971 .WD_Bref {
10972 display:inline !important;
10973 padding:2px !important;
10974 font-size:9pt !important;
10975 width:120px !important;
10976 height:12px !important;
10977 line-height:1.0pt !important;
10978 height:15pt !important;
10979 }
10980
10981 .LiveHtmlCodeviewText {
10982 font-size:10pt;
10983 font-family:Courier New;
10984 xwhite-space:pre;
10985 }
10986
10987 .WD_Panel {
10988 x-index:100 !important;
10989 color:#000 !important;
10990 margin-left:25px !important;
10991 width:800px !important;
10992 padding:4px !important;
10993 border:1px solid #888 !important;
10994 border-radius:6px !important;
10995 background-color:colorrgba(220,220,220,0.9) !important;
10996 font-size:9pt;
10997 font-family:Courier New;
10998 }
10999 .WD_Help {
11000 font-size:10pt !important;
11001 font-family:Courier New;
11002 line-height:1.2 !important;
11003 color:#000 !important;
11004 width:100px !important;
11005 background-color:colorrgba(240,240,255,0.8) !important;
11006 }
11007
11008 .WB_Zoom {
11009 }
11010 </style>
11011 <h2>CosmoScreen 0.0.8</h2>
11012
11013 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
11014 q ... grid on/off<b><br>
11015 i ... zoom in<br>

```

```

11016 o ... zoom out<br>
11017 s ... save current scope<br>
11018 r ... restore saved scope<br>
11019</span>
11020</menu>
11021<div class="WD_Panel" draggable="true">
11022<p><!-- should be on the frame of the WD -->
11023 Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
11024 <input id="WD_Width_1" class="WD_Config" type="text" value="1000">
11025 x <input id="WD_Height_1" class="WD_Config" type="text" value="600">
11026 wall<-paper> <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaper03.png</a>
11027</p>
11028<p>
11029 Desktop <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
11030 | <input id="WD_1_Width" class="WD_Config" type="text" value="0">
11031 x <input id="WD_1_Height" class="WD_Config" type="text" value="0">
11032</p>
11033<p>
11034 Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
11035 <input id="WD_Zoom_1_IN" class="WD_Config" type="text" value="1.0">
11036 | <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="14421356">
11037 <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
11038 <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
11039</p>
11040<p>
11041 Control <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
11042 | <input id="WD_1_In" class="WD_Config" type="text" value="0">
11043 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
11044 shift+wheel for horizontal scroll
11045</p>
11046<p>
11047 Scopexx <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
11048 | <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
11049 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
11050 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
11051</p>
11052<p>
11053 Scopyy <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
11054 <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
11055 <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
11056 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
11057</p>
11058<p>
11059 Popzz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
11060 <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Pop0">
11061 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
11062 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Pop0">
11063</p>
11064<p>
11065 Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
11066</p>
11067<p>
11068 Overflow <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
11069 "scroll" imprisons windows inside the display
11070</p>
11071</div>
11072<div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" style="" contenteditable="true">
11073<div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
11074 <span id="VirtualDesktop_1_Clock"></span>
11075</div>
11076<div id="VirtualDesktop_1_Calender" class="VirtualDesktopCalender" >00:00</div>
11077<div align="right"><h1>VirtualSpace 1.</h1></div>
11078<div id="VirtualDesktop_1_Content" class="VirtualSpace">
11079</div>
11080<div id="VirtualBrowser_1" class="VirtualBrowser" spellcheck="false" draggable="true">
11081<div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar"></div>
11082<span id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar"><reload></span>
11083<iframe id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""/></iframe>
11084</div>
11085<div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11086<div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
11087<span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar"><reload></span>
11088<iframe id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""/></iframe>
11089</div>
11090<div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
11091</div>
11092<div id="VirtualDesktopCodeview" class="VirtualSpace">
11093<input class="HtmlCodeEditButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()"/>
11094 <span id="VirtualDesktopCodeview"></span>
11095 <input id="VirtualDesktopCodeScript">
11096<function vd_showHtmlCode(){
11097   codeSpan = document.getElementById('VirtualDesktopCodeSpan');
11098   showWmCode(WmCode(VirtualDesktopCodeview,codeSpan));
11099   VirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
11100 }
11101<destroyEventSharingCodeview = function(){
11102   VirtualDesktopCodeview.setFunction();
11103   VirtualDesktopCodeview.innerHTML = '';
11104 }
11105<function vdLog(log){
11106   if( GJ_Channel != null ){
11107     GJ_SendMessage('WD '+log);
11108   }
11109   console.log(log);
11110 }
11111<var topMostWin = 10000;
11112<function onEnterWin(e){
11113   if( e.target == topMostWin ) {
11114     t = e.target;
11115     oindex = t.style.zIndex;
11116     //if( oindex == '' ) oindex = 0;
11117     //t.saved.zindex = oindex;
11118     //t.style.zindex = 10000;
11119     t.style.zindex = topMostWin;
11120     nindex = t.style.zindex;
11121     vdLog('Enter '+t.id+' #'+t.id+'('+oindex+'->'+nindex+')');
11122     e.stopPropagation();
11123     e.preventDefault();
11124   }
11125<function onClickWin(e){ // can detect click on the thick border? t = e.target;
11126   oindex = t.style.zIndex;
11127   topMostWin++;
11128   t.style.zindex = topMostWin;
11129   nindex = t.style.zindex;
11130   vdLog('Enter '+t.id+' #'+t.id+'('+oindex+'->'+nindex+')');
11131   e.stopPropagation();
11132   e.preventDefault();
11133 }
11134<function onLeaveWin(e){
11135   if( e.target == topMostWin ) {
11136     t = e.target;
11137     /oindex = t.style.zIndex;
11138     /nindex = t.saved.zindex;
11139     /t.style.zindex = nindex;
11140     vdLog('Click '+t.id+' #'+t.id+'('+oindex+'->'+nindex+')');
11141     /e.stopPropagation();
11142     /e.preventDefault();
11143   }
11144<function onLeaveWin(e){
11145   if( e.target == topMostWin ) {
11146     /oindex = t.style.zIndex;
11147     /nindex = t.saved.zindex;
11148     /t.style.zindex = nindex;
11149     /vdLog('Leave '+e.target.id+' #'+e.target.id+'('+oindex+'->'+nindex+')');
11150     e.stopPropagation();
11151     e.preventDefault();
11152   }
11153<var WinDragstartX; // event
11154 var WinDragstartY;
11155 var WinDragstartT;
11156 var WinDragstartTarget; // target
11157 var WinDragstartTl;
11158<function onWinDragstart(e){
11159   WinDragstartX = e.x;
11160   WinDragstartY = e.y;
11161   WinDragstartT = e.y;
11162   t = e.target;
11163   /t.style.left;
11164   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11165   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11166   if( t.style.left == '' ){
11167     WinDragstartTX = x0 = 0;
11168     t.style.left = '0px';
11169   }else{
11170     /WinDragstartTX = x0 = Number(t.style.left);
11171     WinDragstartTY = x0 = parseInt(t.style.left);
11172   }
11173   if( t.style.top == '' ){
11174     WinDragstartTY = y0 = 0;
11175     t.style.top = '0px';
11176   }else{
11177     /t.style.top = '0px';
11178   }
11179 }
11180<function onWinDragstart(e){
11181   WinDragstartX = e.x;
11182   WinDragstartY = e.y;
11183   WinDragstartT = e.y;
11184   t = e.target;
11185   /t.style.left;
11186   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11187   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11188   if( t.style.left == '' ){
11189     WinDragstartTX = x0 = 0;
11190     t.style.left = '0px';
11191   }else{
11192     /WinDragstartTX = x0 = Number(t.style.left);
11193     WinDragstartTY = x0 = parseInt(t.style.left);
11194   }
11195   if( t.style.top == '' ){
11196     WinDragstartTY = y0 = 0;
11197     t.style.top = '0px';
11198   }else{
11199     /t.style.top = '0px';
11199   }
11200 }
11201<function onWinDragstart(e){
11202   WinDragstartX = e.x;
11203   WinDragstartY = e.y;
11204   WinDragstartT = e.y;
11205   t = e.target;
11206   /t.style.left;
11207   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11208   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11209   if( t.style.left == '' ){
11210     WinDragstartTX = x0 = 0;
11211     t.style.left = '0px';
11212   }else{
11213     /WinDragstartTX = x0 = Number(t.style.left);
11214     WinDragstartTY = x0 = parseInt(t.style.left);
11215   }
11216   if( t.style.top == '' ){
11217     WinDragstartTY = y0 = 0;
11218     t.style.top = '0px';
11219   }else{
11220     /t.style.top = '0px';
11221   }
11222 }
11223<function onWinDragstart(e){
11224   WinDragstartX = e.x;
11225   WinDragstartY = e.y;
11226   WinDragstartT = e.y;
11227   t = e.target;
11228   /t.style.left;
11229   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11230   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11231   if( t.style.left == '' ){
11232     WinDragstartTX = x0 = 0;
11233     t.style.left = '0px';
11234   }else{
11235     /WinDragstartTX = x0 = Number(t.style.left);
11236     WinDragstartTY = x0 = parseInt(t.style.left);
11237   }
11238   if( t.style.top == '' ){
11239     WinDragstartTY = y0 = 0;
11240     t.style.top = '0px';
11241   }else{
11242     /t.style.top = '0px';
11243   }
11244 }
11245<function onWinDragstart(e){
11246   WinDragstartX = e.x;
11247   WinDragstartY = e.y;
11248   WinDragstartT = e.y;
11249   t = e.target;
11250   /t.style.left;
11251   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11252   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11253   if( t.style.left == '' ){
11254     WinDragstartTX = x0 = 0;
11255     t.style.left = '0px';
11256   }else{
11257     /WinDragstartTX = x0 = Number(t.style.left);
11258     WinDragstartTY = x0 = parseInt(t.style.left);
11259   }
11260   if( t.style.top == '' ){
11261     WinDragstartTY = y0 = 0;
11262     t.style.top = '0px';
11263   }else{
11264     /t.style.top = '0px';
11265   }
11266 }
11267<function onWinDragstart(e){
11268   WinDragstartX = e.x;
11269   WinDragstartY = e.y;
11270   WinDragstartT = e.y;
11271   t = e.target;
11272   /t.style.left;
11273   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11274   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11275   if( t.style.left == '' ){
11276     WinDragstartTX = x0 = 0;
11277     t.style.left = '0px';
11278   }else{
11279     /WinDragstartTX = x0 = Number(t.style.left);
11280     WinDragstartTY = x0 = parseInt(t.style.left);
11281   }
11282   if( t.style.top == '' ){
11283     WinDragstartTY = y0 = 0;
11284     t.style.top = '0px';
11285   }else{
11286     /t.style.top = '0px';
11287   }
11288 }
11289<function onWinDragstart(e){
11290   WinDragstartX = e.x;
11291   WinDragstartY = e.y;
11292   WinDragstartT = e.y;
11293   t = e.target;
11294   /t.style.left;
11295   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11296   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11297   if( t.style.left == '' ){
11298     WinDragstartTX = x0 = 0;
11299     t.style.left = '0px';
11300   }else{
11301     /WinDragstartTX = x0 = Number(t.style.left);
11302     WinDragstartTY = x0 = parseInt(t.style.left);
11303   }
11304   if( t.style.top == '' ){
11305     WinDragstartTY = y0 = 0;
11306     t.style.top = '0px';
11307   }else{
11308     /t.style.top = '0px';
11309   }
11310 }
11311<function onWinDragstart(e){
11312   WinDragstartX = e.x;
11313   WinDragstartY = e.y;
11314   WinDragstartT = e.y;
11315   t = e.target;
11316   /t.style.left;
11317   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11318   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11319   if( t.style.left == '' ){
11320     WinDragstartTX = x0 = 0;
11321     t.style.left = '0px';
11322   }else{
11323     /WinDragstartTX = x0 = Number(t.style.left);
11324     WinDragstartTY = x0 = parseInt(t.style.left);
11325   }
11326   if( t.style.top == '' ){
11327     WinDragstartTY = y0 = 0;
11328     t.style.top = '0px';
11329   }else{
11330     /t.style.top = '0px';
11331   }
11332 }
11333<function onWinDragstart(e){
11334   WinDragstartX = e.x;
11335   WinDragstartY = e.y;
11336   WinDragstartT = e.y;
11337   t = e.target;
11338   /t.style.left;
11339   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11340   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11341   if( t.style.left == '' ){
11342     WinDragstartTX = x0 = 0;
11343     t.style.left = '0px';
11344   }else{
11345     /WinDragstartTX = x0 = Number(t.style.left);
11346     WinDragstartTY = x0 = parseInt(t.style.left);
11347   }
11348   if( t.style.top == '' ){
11349     WinDragstartTY = y0 = 0;
11350     t.style.top = '0px';
11351   }else{
11352     /t.style.top = '0px';
11353   }
11354 }
11355<function onWinDragstart(e){
11356   WinDragstartX = e.x;
11357   WinDragstartY = e.y;
11358   WinDragstartT = e.y;
11359   t = e.target;
11360   /t.style.left;
11361   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11362   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11363   if( t.style.left == '' ){
11364     WinDragstartTX = x0 = 0;
11365     t.style.left = '0px';
11366   }else{
11367     /WinDragstartTX = x0 = Number(t.style.left);
11368     WinDragstartTY = x0 = parseInt(t.style.left);
11369   }
11370   if( t.style.top == '' ){
11371     WinDragstartTY = y0 = 0;
11372     t.style.top = '0px';
11373   }else{
11374     /t.style.top = '0px';
11375   }
11376 }
11377<function onWinDragstart(e){
11378   WinDragstartX = e.x;
11379   WinDragstartY = e.y;
11380   WinDragstartT = e.y;
11381   t = e.target;
11382   /t.style.left;
11383   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11384   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11385   if( t.style.left == '' ){
11386     WinDragstartTX = x0 = 0;
11387     t.style.left = '0px';
11388   }else{
11389     /WinDragstartTX = x0 = Number(t.style.left);
11390     WinDragstartTY = x0 = parseInt(t.style.left);
11391   }
11392   if( t.style.top == '' ){
11393     WinDragstartTY = y0 = 0;
11394     t.style.top = '0px';
11395   }else{
11396     /t.style.top = '0px';
11397   }
11398 }
11399<function onWinDragstart(e){
11400   WinDragstartX = e.x;
11401   WinDragstartY = e.y;
11402   WinDragstartT = e.y;
11403   t = e.target;
11404   /t.style.left;
11405   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11406   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11407   if( t.style.left == '' ){
11408     WinDragstartTX = x0 = 0;
11409     t.style.left = '0px';
11410   }else{
11411     /WinDragstartTX = x0 = Number(t.style.left);
11412     WinDragstartTY = x0 = parseInt(t.style.left);
11413   }
11414   if( t.style.top == '' ){
11415     WinDragstartTY = y0 = 0;
11416     t.style.top = '0px';
11417   }else{
11418     /t.style.top = '0px';
11419   }
11420 }
11421<function onWinDragstart(e){
11422   WinDragstartX = e.x;
11423   WinDragstartY = e.y;
11424   WinDragstartT = e.y;
11425   t = e.target;
11426   /t.style.left;
11427   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11428   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11429   if( t.style.left == '' ){
11430     WinDragstartTX = x0 = 0;
11431     t.style.left = '0px';
11432   }else{
11433     /WinDragstartTX = x0 = Number(t.style.left);
11434     WinDragstartTY = x0 = parseInt(t.style.left);
11435   }
11436   if( t.style.top == '' ){
11437     WinDragstartTY = y0 = 0;
11438     t.style.top = '0px';
11439   }else{
11440     /t.style.top = '0px';
11441   }
11442 }
11443<function onWinDragstart(e){
11444   WinDragstartX = e.x;
11445   WinDragstartY = e.y;
11446   WinDragstartT = e.y;
11447   t = e.target;
11448   /t.style.left;
11449   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11450   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11451   if( t.style.left == '' ){
11452     WinDragstartTX = x0 = 0;
11453     t.style.left = '0px';
11454   }else{
11455     /WinDragstartTX = x0 = Number(t.style.left);
11456     WinDragstartTY = x0 = parseInt(t.style.left);
11457   }
11458   if( t.style.top == '' ){
11459     WinDragstartTY = y0 = 0;
11460     t.style.top = '0px';
11461   }else{
11462     /t.style.top = '0px';
11463   }
11464 }
11465<function onWinDragstart(e){
11466   WinDragstartX = e.x;
11467   WinDragstartY = e.y;
11468   WinDragstartT = e.y;
11469   t = e.target;
11470   /t.style.left;
11471   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11472   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11473   if( t.style.left == '' ){
11474     WinDragstartTX = x0 = 0;
11475     t.style.left = '0px';
11476   }else{
11477     /WinDragstartTX = x0 = Number(t.style.left);
11478     WinDragstartTY = x0 = parseInt(t.style.left);
11479   }
11480   if( t.style.top == '' ){
11481     WinDragstartTY = y0 = 0;
11482     t.style.top = '0px';
11483   }else{
11484     /t.style.top = '0px';
11485   }
11486 }
11487<function onWinDragstart(e){
11488   WinDragstartX = e.x;
11489   WinDragstartY = e.y;
11490   WinDragstartT = e.y;
11491   t = e.target;
11492   /t.style.left;
11493   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11494   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11495   if( t.style.left == '' ){
11496     WinDragstartTX = x0 = 0;
11497     t.style.left = '0px';
11498   }else{
11499     /WinDragstartTX = x0 = Number(t.style.left);
11500     WinDragstartTY = x0 = parseInt(t.style.left);
11501   }
11502   if( t.style.top == '' ){
11503     WinDragstartTY = y0 = 0;
11504     t.style.top = '0px';
11505   }else{
11506     /t.style.top = '0px';
11507   }
11508 }
11509<function onWinDragstart(e){
11510   WinDragstartX = e.x;
11511   WinDragstartY = e.y;
11512   WinDragstartT = e.y;
11513   t = e.target;
11514   /t.style.left;
11515   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11516   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11517   if( t.style.left == '' ){
11518     WinDragstartTX = x0 = 0;
11519     t.style.left = '0px';
11520   }else{
11521     /WinDragstartTX = x0 = Number(t.style.left);
11522     WinDragstartTY = x0 = parseInt(t.style.left);
11523   }
11524   if( t.style.top == '' ){
11525     WinDragstartTY = y0 = 0;
11526     t.style.top = '0px';
11527   }else{
11528     /t.style.top = '0px';
11529   }
11530 }
11531<function onWinDragstart(e){
11532   WinDragstartX = e.x;
11533   WinDragstartY = e.y;
11534   WinDragstartT = e.y;
11535   t = e.target;
11536   /t.style.left;
11537   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11538   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11539   if( t.style.left == '' ){
11540     WinDragstartTX = x0 = 0;
11541     t.style.left = '0px';
11542   }else{
11543     /WinDragstartTX = x0 = Number(t.style.left);
11544     WinDragstartTY = x0 = parseInt(t.style.left);
11545   }
11546   if( t.style.top == '' ){
11547     WinDragstartTY = y0 = 0;
11548     t.style.top = '0px';
11549   }else{
11550     /t.style.top = '0px';
11551   }
11552 }
11553<function onWinDragstart(e){
11554   WinDragstartX = e.x;
11555   WinDragstartY = e.y;
11556   WinDragstartT = e.y;
11557   t = e.target;
11558   /t.style.left;
11559   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11560   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11561   if( t.style.left == '' ){
11562     WinDragstartTX = x0 = 0;
11563     t.style.left = '0px';
11564   }else{
11565     /WinDragstartTX = x0 = Number(t.style.left);
11566     WinDragstartTY = x0 = parseInt(t.style.left);
11567   }
11568   if( t.style.top == '' ){
11569     WinDragstartTY = y0 = 0;
11570     t.style.top = '0px';
11571   }else{
11572     /t.style.top = '0px';
11573   }
11574 }
11575<function onWinDragstart(e){
11576   WinDragstartX = e.x;
11577   WinDragstartY = e.y;
11578   WinDragstartT = e.y;
11579   t = e.target;
11580   /t.style.left;
11581   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11582   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11583   if( t.style.left == '' ){
11584     WinDragstartTX = x0 = 0;
11585     t.style.left = '0px';
11586   }else{
11587     /WinDragstartTX = x0 = Number(t.style.left);
11588     WinDragstartTY = x0 = parseInt(t.style.left);
11589   }
11590   if( t.style.top == '' ){
11591     WinDragstartTY = y0 = 0;
11592     t.style.top = '0px';
11593   }else{
11594     /t.style.top = '0px';
11595   }
11596 }
11597<function onWinDragstart(e){
11598   WinDragstartX = e.x;
11599   WinDragstartY = e.y;
11600   WinDragstartT = e.y;
11601   t = e.target;
11602   /t.style.left;
11603   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11604   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11605   if( t.style.left == '' ){
11606     WinDragstartTX = x0 = 0;
11607     t.style.left = '0px';
11608   }else{
11609     /WinDragstartTX = x0 = Number(t.style.left);
11610     WinDragstartTY = x0 = parseInt(t.style.left);
11611   }
11612   if( t.style.top == '' ){
11613     WinDragstartTY = y0 = 0;
11614     t.style.top = '0px';
11615   }else{
11616     /t.style.top = '0px';
11617   }
11618 }
11619<function onWinDragstart(e){
11620   WinDragstartX = e.x;
11621   WinDragstartY = e.y;
11622   WinDragstartT = e.y;
11623   t = e.target;
11624   /t.style.left;
11625   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11626   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11627   if( t.style.left == '' ){
11628     WinDragstartTX = x0 = 0;
11629     t.style.left = '0px';
11630   }else{
11631     /WinDragstartTX = x0 = Number(t.style.left);
11632     WinDragstartTY = x0 = parseInt(t.style.left);
11633   }
11634   if( t.style.top == '' ){
11635     WinDragstartTY = y0 = 0;
11636     t.style.top = '0px';
11637   }else{
11638     /t.style.top = '0px';
11639   }
11640 }
11641<function onWinDragstart(e){
11642   WinDragstartX = e.x;
11643   WinDragstartY = e.y;
11644   WinDragstartT = e.y;
11645   t = e.target;
11646   /t.style.left;
11647   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11648   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11649   if( t.style.left == '' ){
11650     WinDragstartTX = x0 = 0;
11651     t.style.left = '0px';
11652   }else{
11653     /WinDragstartTX = x0 = Number(t.style.left);
11654     WinDragstartTY = x0 = parseInt(t.style.left);
11655   }
11656   if( t.style.top == '' ){
11657     WinDragstartTY = y0 = 0;
11658     t.style.top = '0px';
11659   }else{
11660     /t.style.top = '0px';
11661   }
11662 }
11663<function onWinDragstart(e){
11664   WinDragstartX = e.x;
11665   WinDragstartY = e.y;
11666   WinDragstartT = e.y;
11667   t = e.target;
11668   /t.style.left;
11669   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11670   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11671   if( t.style.left == '' ){
11672     WinDragstartTX = x0 = 0;
11673     t.style.left = '0px';
11674   }else{
11675     /WinDragstartTX = x0 = Number(t.style.left);
11676     WinDragstartTY = x0 = parseInt(t.style.left);
11677   }
11678   if( t.style.top == '' ){
11679     WinDragstartTY = y0 = 0;
11680     t.style.top = '0px';
11681   }else{
11682     /t.style.top = '0px';
11683   }
11684 }
11685<function onWinDragstart(e){
11686   WinDragstartX = e.x;
11687   WinDragstartY = e.y;
11688   WinDragstartT = e.y;
11689   t = e.target;
11690   /t.style.left;
11691   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11692   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11693   if( t.style.left == '' ){
11694     WinDragstartTX = x0 = 0;
11695     t.style.left = '0px';
11696   }else{
11697     /WinDragstartTX = x0 = Number(t.style.left);
11698     WinDragstartTY = x0 = parseInt(t.style.left);
11699   }
11700   if( t.style.top == '' ){
11701     WinDragstartTY = y0 = 0;
11702     t.style.top = '0px';
11703   }else{
11704     /t.style.top = '0px';
11705   }
11706 }
11707<function onWinDragstart(e){
11708   WinDragstartX = e.x;
11709   WinDragstartY = e.y;
11710   WinDragstartT = e.y;
11711   t = e.target;
11712   /t.style.left;
11713   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11714   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11715   if( t.style.left == '' ){
11716     WinDragstartTX = x0 = 0;
11717     t.style.left = '0px';
11718   }else{
11719     /WinDragstartTX = x0 = Number(t.style.left);
11720     WinDragstartTY = x0 = parseInt(t.style.left);
11721   }
11722   if( t.style.top == '' ){
11723     WinDragstartTY = y0 = 0;
11724     t.style.top = '0px';
11725   }else{
11726     /t.style.top = '0px';
11727   }
11728 }
11729<function onWinDragstart(e){
11730   WinDragstartX = e.x;
11731   WinDragstartY = e.y;
11732   WinDragstartT = e.y;
11733   t = e.target;
11734   /t.style.left;
11735   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11736   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11737   if( t.style.left == '' ){
11738     WinDragstartTX = x0 = 0;
11739     t.style.left = '0px';
11740   }else{
11741     /WinDragstartTX = x0 = Number(t.style.left);
11742     WinDragstartTY = x0 = parseInt(t.style.left);
11743   }
11744   if( t.style.top == '' ){
11745     WinDragstartTY = y0 = 0;
11746     t.style.top = '0px';
11747   }else{
11748     /t.style.top = '0px';
11749   }
11750 }
11751<function onWinDragstart(e){
11752   WinDragstartX = e.x;
11753   WinDragstartY = e.y;
11754   WinDragstartT = e.y;
11755   t = e.target;
11756   /t.style.left;
11757   /WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11758   /WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11759   if( t.style.left == '' ){
11760    
```

```

11178     //WinDragstartXY = y0 = Number(t.style.top);
11179     WinDragstartXY = y0 = parseInt(t.style.top);
11180 }
11181 if( true ){ // to be undo
11182     t.wasAtX = WinDragstartTX;
11183     t.wasAtY = WinDragstartTY;
11184 }
11185 wlog('DragTA #' + t.id
11186     + ' event: '+e.clientX+'x'
11187     + ' position: '+t.style.position
11188     + ' style left,top('+t.style.left+','+t.style.top+')');
11189 };
11190 e.stopPropagation();
11191 //e.preventDefault();
11192 return true;
11193}
11194 function onWinDragEvent(wh,e,set,dolog){
11195     t = e.target;
11196     dx = e.x - WinDragstartX;
11197     dy = e.y - WinDragstartY;
11198     nx = WinDragstartX+dx;
11199     ny = WinDragstartY+dy;
11200     log = 'Drag'+wh+' #' + t.id
11201     + ' event0('+WinDragstartX+','+WinDragstartY+')'
11202     + ' event1('+e.clientX+', '+e.clientY)'
11203     + ' diff: '+dx+', '+dy+
11204     + ' nx: '+nx+', '+ny+
11205     + ' (' + t.style.left + ',' + t.style.top + ')'
11206     + ' wasAt(' + t.wasAtX + ', ' + t.wasAtY + ')';
11207 }
11208 if( e.x != 0 || e.y != 0 ){
11209     if( set == true ){
11210         //t.style.x = nx + 'px'; // not effective
11211         //t.style.y = ny + 'px'; // not effective
11212         t.style.left = nx + 'px';
11213         t.style.top = ny + 'px';
11214         log += ' Set';
11215     }else{
11216         log += ' NotSet';
11217         if( !dolog ){
11218             log = '';
11219         }
11220     }
11221 }else{
11222     log += ' What?'; // the type is event start?
11223     if( !dolog ){
11224         log = '';
11225     }
11226 }
11227 if( and(dolog, log != '') ){
11228     wlog(log);
11229 }
11230 if( true ){
11231     // should be propigated to parent in FireFox ?
11232     e.stopPropagation();
11233 }
11234 e.preventDefault();
11235 return false;
11236}
11237 function onWinDrag(e){
11238     return onWinDragEvent('Ing',e,true,false);
11239}
11240 function onWinDragend(e){
11241     return onWinDragEvent('End',e,false,true);
11242}
11243 function onWinDragexit(e){
11244     return onWinDragEvent('Exit',e,false,true);
11245}
11246 function onWinDragover(e){
11247     return onWinDragEvent('Over',e,false,true);
11248}
11249 function onWinDragenter(e){
11250     return onWinDragEvent('Enter',e,false,true);
11251}
11252 function onWinDragleave(e){
11253     return onWinDragEvent('Leave',e,false,true);
11254}
11255 function onWinDragdrop(e){
11256     return onWinDragEvent('Drop',e,false,true);
11257}
11258 function onFaviconChange(e){
11259     wlog('---Favicon #' + e.target.id + ' href=' + e.details.href);
11260}
11261 var savedSuppressGJShell = false;
11262 function stopShell(e){
11263     //wlog('enter Gsh STOP');
11264     savedSuppressGJShell = SuppressGJShell;
11265     SuppressGJShell = true;
11266     e.stopPropagation();
11267     e.preventDefault();
11268}
11269 function contShell(e){
11270     //wlog('leave Gsh STOP');
11271     SuppressGJShell = savedSuppressGJShell;
11272     e.stopPropagation();
11273     e.preventDefault();
11274}
11275
11276 function WD_onkeydown(e){
11277     keycode = e.keyCode;
11278     cmdos1.log('keydown #' + e.target.id + ' ' + keycode);
11279     if( keycode == 'KeyG' ){
11280         WD_SetGrid1(WD_Grid_1);
11281     }else{
11282         if( keycode == 'KeyI' ){
11283             WD_DoZoomIn();
11284         }else{
11285             if( keycode == 'KeyO' ){
11286                 WD_DoZoomOut();
11287             }else{
11288                 if( keycode == 'KeyR' ){
11289                     WD_RestoreScope(null);
11290                 }else{
11291                     if( keycode == 'KeyS' ){
11292                         WD_SaveScope(null);
11293                     }
11294                 }
11295             }
11296         }
11297     }
11298     e.stopPropagation();
11299     e.preventDefault();
11300}
11301 function WD_EventSetup(){
11302     VirtualDesktop_1.addEventListener('keydown', e => { WD_onkeydown(e); });
11303     VirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onkeydown(e); });
11304     Help_1.addEventListener('keydown', e => { WD_onkeydown(e); });
11305     Help_1.addEventListener('keyup', e => { WD_onkeyup(e); });
11306}
11307 function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
11308     function VirtualBrowserCommand(e,s,l,cmd,f){
11309         command = cmd.innerHTML;
11310         if( command == "Reload" ){
11311             href_id = e.target.href_id;
11312             d = document.getElementById(href_id);
11313             wlog('href tag '+href_id+'\n elem#'+href_id+'\n href=' + d);
11314             url = d.innerHTML;
11315             wlog('---- Load href_tag#' + href_id + '\n elem#' + href_id + '\n href=' + d
11316             + '\n url=' + url);
11317             wlog('---- Load target #' + f.id + ' with url=' + url);
11318             f.src = url;
11319         }else{
11320             alert('unknown command' + command + ' ' + e.target.id + ', ' + l.id + ', ' + f.id);
11321         }
11322     }
11323     function onkeyDown(e){
11324         if( e.code == 'Enter' ){
11325             e.stopPropagation();
11326             e.preventDefault();
11327         }
11328     }
11329     function onkeyUp(e){
11330         if( e.code == 'Enter' ){
11331             e.stopPropagation();
11332             e.preventDefault();
11333             // should reload immediately ?
11334         }
11335     }
11336     if( false ){
11337         wlog('start settle VirtualBrowser url=' + u + '\n'
11338             + 'id=' + s.id + '\n');
11339     }

```

```

11340     + 'width=' + s.style.width + '\n'
11341     + 'height=' + s.style.height
11342   );
11343   // very important for WordPress ??
11344   s.style.width = f.style.width = 501; // for WordPress ...??
11345   s.style.height = f.style.height = 271; // for WordPress ...??
11346   if( false ){
11347     wlog("midway settle VirtualBrowser url='"+u+'\n'
11348       + 'id=' + s.id + '\n'
11349       + 'width=' + s.style.width + '\n'
11350       + 'height=' + s.style.height
11351   );
11352 }
11353 },width = 502; // for WordPress ...??
11354 s.height = 272; // for WordPress ...??
11355 if( false ){
11356   wlog("midway-2 settle Virtualbrowser url='"+u+'\n'
11357     + 'id=' + s.id + '\n'
11358     + 'span-width=' + s.width + '\n'
11359     + 'span-height=' + s.height
11360   );
11361 }
11362 }
11363 s.style.width = w + 'px';
11364 s.style.height = h + 'px';
11365 s.style.width = w + 'px';
11366 s.style.height = h + 'px';
11367 f.style.setProperty('webkit-transform','scale('+scale+')');
11368 f.style.setProperty('transform','scale('+scale+')');
11369 f.setAttribute('draggable','true');
11370 f.setAttribute('draggable','false'); // why necessary?
11371 l.setAttribute('draggable','false'); // why necessary?
11372 c.addEventListener('dragstart', e => { onWinDragstart(e); });
11373 s.addEventListener('drag', e => { onWinDrag(e); });
11374 s.addEventListener('exit', e => { onWinDragexit(e); });
11375 s.addEventListener('dragend', e => { onWinDragend(e); });
11376 s.addEventListener('dragexit', e => { onWinDragexit(e); });
11377 s.addEventListener('dragover', e => { onWinDragover(e); });
11378 s.addEventListener('dragleave', e => { onWinDragleave(e); });
11379 s.addEventListener('drop', e => { onWinDragdrop(e); });
11380 s.addEventListener('mouseenter',e => { onEnterWin(e); });
11381 s.addEventListener('mouseleave',e => { onLeaveWin(e); });
11382 if( false ){
11383   s.style.position = "absolute";
11384   s.style.x = x+'px';
11385   s.style.left = x+'px';
11386   s.style.y = y+'px';
11387   s.style.top = y+'px';
11388 }else{
11389   s.style.setProperty('position','absolute','important');
11390   s.style.setProperty('x',x+'px','important');
11391   s.style.setProperty('left',x+'px','important');
11392   s.style.setProperty('y',y+'px','important');
11393   s.style.setProperty('top',y+'px','important');
11394 }
11395 favicon = './favicon.ico';
11396 uvl = u.split('/');
11397 if( 2 <= uvl.length ){
11398   uv2 = uvl[1].split('/');
11399   if( 2 <= uv2.length ){
11400     if( uv1[0] == 'file' ){
11401       //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/')
11402       // + '/favicon.ico';
11403       favicon = './favicon.ico';
11404     }else{
11405       favicon = uvl[0] + '//' + uv2[0] + '/favicon.ico';
11406     }
11407   }
11408 },wlog("----- favicon-url=" +favicon);
11409 href_id = 1.id + ' href';
11410 l.innerHTML = '';
11411   + '<a id="'+href_id+'" class="VirtualBrowserLocation" href="'+u+'">' +u+ '</a>';
11412 //l.addEventListener('click',e => { onClickWin(e); });
11413 l.addEventListener('enter',e => { onEnterWin(e); });
11414 l.addEventListener('mouseleave',e => { onLeaveWin(e); });
11415 l.addEventListener('keydown',e => { onKeyDown(e); });
11416 l.addEventListener('keyup',e => { onKeyUp(e); });
11417
11418 cmd.href_id = href_id;
11419 wlog('()cmd#'+cmd.id);
11420 wdlog('()href_id#'+href_id);
11421 wdlog('()2.href_id#'+cmd.href_id);
11422 cmd.addEventListener('click',e => { VirtualBrowserCommand(e,s,l,cmd,f); });
11423
11424 f.style.borderColor = c;
11425 f.arc = u;
11426 //f.addEventListener('mouseenter',e => { onEnterWin(e); });
11427 //f.addEventListener('mouseleave',e => { onLeaveWin(e); });
11428
11429 //s.addEventListener('click', e => { onClickWin(e); });
11430 //f.addEventListener('click', e => { wdlog('click whl'); });
11431 f.addEventListener('mozbrowsericonchange',onFaviconChange);
11432
11433 wlog('done settle VirtualBrowser url=' +u +'\n'
11434   + 'id=' + s.id + ' '
11435   + 'width=' + s.style.width + ' '
11436   + 'height=' + s.style.height + ' '
11437   + 'cmd=' + cmd.id
11438 );
11439
11440
11441 function WD_EventSetup(){
11442   dt = VirtualDesktop1;
11443   dt.style.width = "800px";
11444   dt.style.height = "500px";
11445   dt.addEventListener('dragstart',e => { onWinDragstart(e); });
11446   dt.addEventListener('drag', e => { onWinDrag(e); });
11447   dt.addEventListener('exit', e => { onWinDragexit(e); });
11448 }
11449
11450
11451 function GRonClick(){
11452   WD_SaveScope(null); // should be push
11453   t = event.target;
11454   x = t.getAttribute('data-leftx');
11455   y = t.getAttribute('data-copy');
11456   zoom = WD_Zoom_1_XY.value;
11457   zoom *= zoom;
11458   y *= zoom;
11459   WD_doScrollXY(event,x,y);
11460   //alert('scroll #' +t.id+ ' x=' +x+ ', y=' +y);
11461 }
11462
11463 function WD_setGrid(e){
11464   t = e.target;
11465   WD_SetGrid(t);
11466 }
11467
11468 function WD_setGrid(t){
11469   //ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11470   ds = VirtualDesktop_1_GridPlane;
11471   if( t.value == 'GridOn' ){
11472     for( col = 0; col < 16; col++ ){
11473       for( row = 0; row < 16; row++ ){
11474         g1 = document.createElement('span');
11475         g1.setAttribute('class','VirtualGrid');
11476         leftx = col * 800;
11477         topy = row * 500;
11478         g1 = col + ' ' + row;
11479         label = 'id=' +t.id+ ' ' + 'class="WD_GridScroll" '
11480           + 'contenteditable="false" onclick="GRonClick()" '
11481           + 'data-leftx=' + leftx + ' ' + 'data-topy=' + topy + ' '
11482           + '>';
11483         g1 = g1 + ' ' + 'label';
11484         g1.innerHTML = label;
11485         g1.position = 'relative';
11486         g1.left = leftx;
11487         g1.top = topy;
11488         g1.style.left = g1.leftx + 'px';
11489         g1.style.top = g1.topy + 'px';
11490         if( col % 2 == row % 2 ){
11491           g1.style.backgroundColor = 'rgba(255,255,255,0.3)';
11492         }
11493       }
11494     }
11495   }
11496 }
11497
11498
11499
11500
11501
11502
11503
11504
11505
11506
11507
11508
11509
11510
11511
11512
11513
11514
11515
11516
11517
11518
11519
11520
11521
11522
11523
11524
11525
11526
11527
11528
11529
11530
11531
11532
11533
11534
11535
11536
11537
11538
11539
11540
11541
11542
11543
11544
11545
11546
11547
11548
11549
11550
11551
11552
11553
11554
11555
11556
11557
11558
11559
11560
11561
11562
11563
11564
11565
11566
11567
11568
11569
11570
11571
11572
11573
11574
11575
11576
11577
11578
11579
11580
11581
11582
11583
11584
11585
11586
11587
11588
11589
11590
11591
11592
11593
11594
11595
11596
11597
11598
11599
11600
11601
11602
11603
11604
11605
11606
11607
11608
11609
11610
11611
11612
11613
11614
11615
11616
11617
11618
11619
11620
11621
11622
11623
11624
11625
11626
11627
11628
11629
11630
11631
11632
11633
11634
11635
11636
11637
11638
11639
11640
11641
11642
11643
11644
11645
11646
11647
11648
11649
11650
11651
11652
11653
11654
11655
11656
11657
11658
11659
11660
11661
11662
11663
11664
11665
11666
11667
11668
11669
11670
11671
11672
11673
11674
11675
11676
11677
11678
11679
11680
11681
11682
11683
11684
11685
11686
11687
11688
11689
11690
11691
11692
11693
11694
11695
11696
11697
11698
11699
11700
11701
11702
11703
11704
11705
11706
11707
11708
11709
11710
11711
11712
11713
11714
11715
11716
11717
11718
11719
11720
11721
11722
11723
11724
11725
11726
11727
11728
11729
11730
11731
11732
11733
11734
11735
11736
11737
11738
11739
11740
11741
11742
11743
11744
11745
11746
11747
11748
11749
11750
11751
11752
11753
11754
11755
11756
11757
11758
11759
11760
11761
11762
11763
11764
11765
11766
11767
11768
11769
11770
11771
11772
11773
11774
11775
11776
11777
11778
11779
11780
11781
11782
11783
11784
11785
11786
11787
11788
11789
11790
11791
11792
11793
11794
11795
11796
11797
11798
11799
11800
11801
11802
11803
11804
11805
11806
11807
11808
11809
11810
11811
11812
11813
11814
11815
11816
11817
11818
11819
11820
11821
11822
11823
11824
11825
11826
11827
11828
11829
11830
11831
11832
11833
11834
11835
11836
11837
11838
11839
11840
11841
11842
11843
11844
11845
11846
11847
11848
11849
11850
11851
11852
11853
11854
11855
11856
11857
11858
11859
11860
11861
11862
11863
11864
11865
11866
11867
11868
11869
11870
11871
11872
11873
11874
11875
11876
11877
11878
11879
11880
11881
11882
11883
11884
11885
11886
11887
11888
11889
11890
11891
11892
11893
11894
11895
11896
11897
11898
11899
11900
11901
11902
11903
11904
11905
11906
11907
11908
11909
11910
11911
11912
11913
11914
11915
11916
11917
11918
11919
11920
11921
11922
11923
11924
11925
11926
11927
11928
11929
11930
11931
11932
11933
11934
11935
11936
11937
11938
11939
11940
11941
11942
11943
11944
11945
11946
11947
11948
11949
11950
11951
11952
11953
11954
11955
11956
11957
11958
11959
11960
11961
11962
11963
11964
11965
11966
11967
11968
11969
11970
11971
11972
11973
11974
11975
11976
11977
11978
11979
11980
11981
11982
11983
11984
11985
11986
11987
11988
11989
11990
11991
11992
11993
11994
11995
11996
11997
11998
11999
11999
12000
12001
12002
12003
12004
12005
12006
12007
12008
12009
120010
120011
120012
120013
120014
120015
120016
120017
120018
120019
120020
120021
120022
120023
120024
120025
120026
120027
120028
120029
120030
120031
120032
120033
120034
120035
120036
120037
120038
120039
120040
120041
120042
120043
120044
120045
120046
120047
120048
120049
120050
120051
120052
120053
120054
120055
120056
120057
120058
120059
120060
120061
120062
120063
120064
120065
120066
120067
120068
120069
120070
120071
120072
120073
120074
120075
120076
120077
120078
120079
120080
120081
120082
120083
120084
120085
120086
120087
120088
120089
120090
120091
120092
120093
120094
120095
120096
120097
120098
120099
120099
1200100
1200101
1200102
1200103
1200104
1200105
1200106
1200107
1200108
1200109
1200110
1200111
1200112
1200113
1200114
1200115
1200116
1200117
1200118
1200119
1200120
1200121
1200122
1200123
1200124
1200125
1200126
1200127
1200128
1200129
1200130
1200131
1200132
1200133
1200134
1200135
1200136
1200137
1200138
1200139
1200140
1200141
1200142
1200143
1200144
1200145
1200146
1200147
1200148
1200149
1200150
1200151
1200152
1200153
1200154
1200155
1200156
1200157
1200158
1200159
1200160
1200161
1200162
1200163
1200164
1200165
1200166
1200167
1200168
1200169
1200170
1200171
1200172
1200173
1200174
1200175
1200176
1200177
1200178
1200179
1200180
1200181
1200182
1200183
1200184
1200185
1200186
1200187
1200188
1200189
1200190
1200191
1200192
1200193
1200194
1200195
1200196
1200197
1200198
1200199
1200199
1200200
1200201
1200202
1200203
1200204
1200205
1200206
1200207
1200208
1200209
1200210
1200211
1200212
1200213
1200214
1200215
1200216
1200217
1200218
1200219
1200219
1200220
1200221
1200222
1200223
1200224
1200225
1200226
1200227
1200228
1200229
1200229
1200230
1200231
1200232
1200233
1200234
1200235
1200236
1200237
1200238
1200239
1200239
1200240
1200241
1200242
1200243
1200244
1200245
1200246
1200247
1200248
1200249
1200249
1200250
1200251
1200252
1200253
1200254
1200255
1200256
1200257
1200258
1200259
1200259
1200260
1200261
1200262
1200263
1200264
1200265
1200266
1200267
1200268
1200269
1200269
1200270
1200271
1200272
1200273
1200274
1200275
1200276
1200277
1200278
1200279
1200279
1200280
1200281
1200282
1200283
1200284
1200285
1200286
1200287
1200288
1200289
1200289
1200290
1200291
1200292
1200293
1200294
1200295
1200296
1200297
1200298
1200299
1200299
1200300
1200301
1200302
1200303
1200304
1200305
1200306
1200307
1200308
1200309
1200309
1200310
1200311
1200312
1200313
1200314
1200315
1200316
1200317
1200318
1200319
1200319
1200320
1200321
1200322
1200323
1200324
1200325
1200326
1200327
1200328
1200329
1200329
1200330
1200331
1200332
1200333
1200334
1200335
1200336
1200337
1200338
1200339
1200339
1200340
1200341
1200342
1200343
1200344
1200345
1200346
1200347
1200348
1200349
1200349
1200350
1200351
1200352
1200353
1200354
1200355
1200356
1200357
1200358
1200359
1200359
1200360
1200361
1200362
1200363
1200364
1200365
1200366
1200367
1200368
1200369
1200369
1200370
1200371
1200372
1200373
1200374
1200375
1200376
1200377
1200378
1200379
1200379
1200380
1200381
1200382
1200383
1200384
1200385
1200386
1200387
1200388
1200389
1200389
1200390
1200391
1200392
1200393
1200394
1200395
1200396
1200397
1200398
1200399
1200399
1200400
1200401
1200402
1200403
1200404
1200405
1200406
1200407
1200408
1200409
1200409
1200410
1200411
1200412
1200413
1200414
1200415
1200416
1200417
1200418
1200419
1200419
1200420
1200421
1200422
1200423
1200424
1200425
1200426
1200427
1200428
1200429
1200429
1200430
1200431
1200432
1200433
1200434
1200435
1200436
1200437
1200438
1200439
1200439
1200440
1200441
1200442
1200443
1200444
1200445
1200446
1200447
1200448
1200449
1200449
1200450
1200451
1200452
1200453
1200454
1200455
1200456
1200457
1200458
1200459
1200459
1200460
1200461
1200462
1200463
1200464
1200465
1200466
1200467
1200468
1200469
1200469
1200470
1200471
1200472
1200473
1200474
1200475
1200476
1200477
1200478
1200479
1200479
1200480
1200481
1200482
1200483
1200484
1200485
1200486
1200487
1200488
1200489
1200489
1200490
1200491
1200492
1200493
1200494
1200495
1200496
1200497
1200498
1200499
1200499
1200500
1200501
1200502
1200503
1200504
1200505
1200506
1200507
1200508
1200509
1200509
1200510
1200511
1200512
1200513
1200514
1200515
1200516
1200517
1200518
1200519
1200519
1200520
1200521
1200522
1200523
1200524
1200525
1200526
1200527
1200528
1200529
1200529
1200530
1200531
1200532
1200533
1200534
1200535
1200536
1200537
1200538
1200539
1200539
1200540
1200541
1200542
1200543
1200544
1200545
1200546
1200547
1200548
1200549
1200549
1200550
1200551
1200552
1200553
1200554
1200555
1200556
1200557
1200558
1200559
1200559
1200560
1200561
1200562
1200563
1200564
1200565
1200566
1200567
1200568
1200569
1200569
1200570
1200571
1200572
1200573
1200574
1200575
1200576
1200577
1200578
1200579
1200579
1200580
1200581
1200582
1200583
1200584
1200585
1200586
1200587
1200588
1200589
1200589
1200590
1200591
1200592
1200593
1200594
1200595
1200596
1200597
1200598
1200599
1200599
1200600
1200601
1200602
1200603
1200604
1200605
1200606
1200607
1200608
1200609
1200609
1200610
1200611
1200612
1200613
1200614
1200615
1200616
1200617
1200618
1200619
1200619
1200620
1200621
1200622
1200623
1200624
1200625
1200626
1200627
1200628
1200629
1200629
1200630
1200631
1200632
1200633
1200634
1200635
1200636
1200637
1200638
1200639
1200639
1200640
1200641
1200642
1200643
1200644
1200645
1200646
1200647
1200648
1200649
1200649
1200650
1200651
1200652
1200653
1200654
1200655
1200656
1200657
1200658
1200659
1200659
1200660
1200661
1200662
1200663
1200664
1200665
1200666
1200667
1200668
1200669
120066
```

```

11502         ds.appendChild(g1);
11503     }
11504   }
11505   t.value = 'Gridoff';
11506 } else{
11507   ds.innerHTML = '';
11508   t.value = 'GridOn';
11509 }
11510 }
11511
11512 var sav_scrollLeft;
11513 var sav_scrollTop;
11514 var sav_nscale;
11515 function SavScope(e){
11516   sav_scrollLeft = WD_left_1.value;
11517   sav_scrollTop = WD_top_1.value;
11518   sav_nscale = WD_Zoom_1_XY.value;
11519   //console.log('saved zoom=' +sav_nscale);
11520 }
11521 function WD_RestoreScope(e){
11522   WD_Zoom_1_XY.value = sav_nscale;
11523   WD_doZoom();
11524 }
11525 WD_left_1.value = sav_scrollLeft;
11526 WD_top_1.value = sav_scrollTop;
11527 WD_doScroll(null);
11528 }
11529 function ignoreEvent(e){
11530   e.stopPropagation();
11531   //e.preventDefault();
11532 }
11533 function zoomMag(){
11534   return WD_Zoom_1_MAG.value;
11535 }
11536 function WD_EventSetup3(){
11537   WD_Grid_1.addEventListener('click', e => { WD_SetGrid(e); });
11538   WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
11539   WD_Zoom_1_Command.addEventListener('click', e => { WD_RestoreScope(e); });
11540   WD_Width_1.value = dt.style.width;
11541   WD_Width_1.addEventListener('keydown',ignoreEvent);
11542   WD_Height_1.value = dt.style.height;
11543   WD_Height_1.addEventListener('keyup',ignoreEvent);
11544   WD_Height_1.addEventListener('keydown',ignoreEvent);
11545   WD_Height_1.addEventListener('keyup',ignoreEvent);
11546   WD_Zoom_1_MAG.addEventListener('keydown',ignoreEvent);
11547   WD_Zoom_1_MAG.addEventListener('keyup',ignoreEvent);
11548 }
11549
11550 function escale(e,oscale,nscale){
11551   e.style.setProperty('transform','scale('+nscale+')');
11552   rscale = oscale / nscale;
11553   w = parseint(e.style.width);
11554   h = parseint(e.style.height);
11555   w = w * rscale; //oscale/nscale;
11556   h = h * rscale; //oscale/nscale;
11557   e.style.width = w + 'px';
11558   e.style.height = h + 'px';
11559 }
11560 function scaleWD(ds,oscale,nscale){
11561   if( true ) {
11562     escale(WirtualBrowser_1_Location,oscale,nscale);
11563     escale(WirtualBrowser_1_Command,oscale,nscale);
11564     escale(WirtualBrowser_1_Frame,oscale,nscale);
11565   }
11566   escale(WirtualBrowser_2_Location,oscale,nscale);
11567   escale(WirtualBrowser_2_Command,oscale,nscale);
11568   escale(WirtualBrowser_2_Frame,oscale,nscale);
11569   escale(WirtualBrowser_3_Location,oscale,nscale);
11570   escale(WirtualBrowser_3_Command,oscale,nscale);
11571   escale(WirtualBrowser_3_Frame,oscale,nscale);
11572 }
11573
11574 function WD_doZoom(){
11575   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
11576   oscale = WD_Zoom_1_XY.value; // hidden value for current zoom
11577   nscale = WD_Zoom_1_XY.value;
11578   ds.style.zoom = nscale;
11579   WD_Zoom_1_XY.value = ds.style.zoom;
11580   WD_Zoom_1_XY.value = ds.style.zoom;
11581   scaledWD(ds,oscale,nscale);
11582 }
11583
11584 function WD_EventSetup(){
11585   WD_Zoom_1.addEventListener('click',WD_doZoom);
11586   WD_Zoom_1_XY.addEventListener('keydown',ignoreEvent);
11587   WD_Zoom_1_XY.addEventListener('keyup',ignoreEvent);
11588 }
11589
11590 function WD_doZoomOUT(){
11591   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
11592   oscale = WD_Zoom_1_XY.value;
11593   if( oscale == 0 || oscale == '' ){
11594     oscale = 1;
11595   }
11596   nscale = oscale / zoomMag();
11597   ds.style.zoom = nscale;
11598   WD_Zoom_1_XY.value = ds.style.zoom;
11599   WD_Zoom_1_XY.value = ds.style.zoom;
11600   scaledWD(ds,oscale,nscale);
11601 }
11602
11603 function WD_doZoomIN(){
11604   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
11605   oscale = WD_Zoom_1_XY.value;
11606   if( oscale == 0 || oscale == '' ){
11607     oscale = 1;
11608   }
11609   nscale = oscale * zoomMag();
11610   if( 4 < nscale ){
11611     alert("maybe too large, zoom=' +nscale");
11612     return;
11613   }
11614   ds.style.zoom = nscale;
11615   WD_Zoom_1_XY.value = ds.style.zoom;
11616   WD_Zoom_1_XY.value = ds.style.zoom;
11617   scaledWD(ds,oscale,nscale);
11618 }
11619
11620 function WD_EventSetup(){
11621   WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
11622   WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
11623 }
11624
11625
11626 function WD_doResize(e){
11627   dt = WirtualDesktop_1;
11628   dt.style.width = WD_Width_1.value;
11629   dt.style.height = WD_Height_1.value;
11630   WD_Width_1.value = dt.style.width;
11631   WD_Height_1.value = dt.style.height;
11632 }
11633
11634 WD_Resize_1.addEventListener('click',e => { WD_doResize(e); });
11635
11636 function WD_doRSResize(e){
11637   //alert('Resize Space');
11638   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
11639   ds.style.width = WS_1_Width.value;
11640   ds.style.height = WS_1_Height.value;
11641   WS_1_Width.value = ds.style.width;
11642   WS_1_Height.value = ds.style.height;
11643 }
11644
11645 function WD_EventSetup(){
11646   ds = WirtualDesktop_1_Content; // should be VirtualSpace_1
11647   ds.style.width = '5120px';
11648   ds.style.height = '2880px';
11649   WS_1_Width.value = ds.style.width;
11650   WS_1_Height.value = ds.style.height;
11651   WS_1_Width.addEventListener('keydown',ignoreEvent);
11652   WS_1_Height.addEventListener('keydown',ignoreEvent);
11653   WS_1_Height.addEventListener('keyup',ignoreEvent);
11654   WS_1_Height.addEventListener('click',e => { WD_doRSResize(e); });
11655 }
11656
11657 function WD_doScrollXY(e,sleft,stop){
11658   dt = WirtualDesktop_1;
11659   dt.scrollLeft = sleft;
11660   dt.scrollTop = stop;
11661   WD_left_1.value = dt.scrollLeft;
11662   WD_top_1.value = dt.scrollTop;
11663   console.log('--Scroll #' +dt.id+ ' ('+sleft+','+stop+')');

```

```

11664}
11665    function WD_descroll(e){
11666        /dt = VirtualDesktop_1_Content;
11667        dt = VirtualDesktop_1;
11668        sleft = parseInt(WD_Left_1.value);
11669        stop = parseInt(WD_Top_1.value);
11670        dt.scrollleft = sleft;
11671        dt.scrollTop = stop;
11672        WD_Left_1.value = dt.scrollleft;
11673        WD_Top_1.value = dt.scrollTop;
11674        console.log('--Scroll #' +dt.id+ ' ('+sleft+','+stop+')');
11675    }
11676    function showScrollPosition(){
11677        if( false ){
11678            console.log(
11679                'wstop' + VirtualDesktop_1.style.top + ',' +
11680                'wsx' + VirtualDesktop_1.style.y + ',' +
11681                'wss' + VirtualDesktop_1.scrollTop + ',' +
11682                'wdtop' + VirtualDesktop_1_Content.style.top + ',' +
11683                'wdx' + VirtualDesktop_1_Content.style.y + ',' +
11684                'wds' + VirtualDesktop_1_Content.scrollTop + ',' +
11685            );
11686        WD_Left_1.value = VirtualDesktop_1.scrollLeft;
11687        WD_Top_1.value = VirtualDesktop_1.scrollTop;
11688    }
11689    function evEventSetup(e){
11690        e.preventDefault();
11691        WD_Left_1.addEventListener('click',e => { WD_descroll(e); });
11692        WD_Left_1.addEventListener('keydown',ignoreEvent);
11693        WD_Left_1.addEventListener('keyup',ignoreEvent);
11694        WD_Top_1.addEventListener('keydown',ignoreEvent);
11695        WD_Top_1.addEventListener('keyup',ignoreEvent);
11696    }
11697    function WD_EventSetup(){
11698        VirtualDesktop_1.addEventListener('scroll',showScrollPosition);
11699        VirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
11700    }
11701    if( false ){
11702        w = 1000 + 'px';
11703        dt.style.width = w;
11704        dt.style.height = "300px";
11705        dt.style.resize = "both";
11706        dt.style.borderWidth = '1px' + 'solid';
11707        dt.style.borderRadius = '25' + 'px';
11708        console.log('----- #' +dt.id+ ' style=' +dt.style);
11709        console.log('----- #' +dt.id+ ' width=' +dt.style.width);
11710        console.log('----- #' +dt.id+ ' left=' +dt.style.left);
11711        console.log('----- #' +dt.id+ ' border=' +dt.style.border);
11712    }
11713    function onDTResize(e){
11714        dt = e.target;
11715        h = parseInt(dt.style.height);
11716        dt.style.borderWidth = (h * 0.075) + 'px';
11717        console.log('----- borderWidth=' +dt.style.borderWidth);
11718    }
11719    VirtualDesktopDetails.addEventListener('toggle',VirtualDesktop_init);
11720    function VirtualDesktop_init(){
11721        if( !VirtualDesktopDetails.open ){
11722            return;
11723        }
11724        //GJ Join;
11725        VirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
11726        //console.log('----- #' +dt.id
11727        // + ' borderWidth=' +dt.style.getProperty('border-width'));
11728    }
11729    settleWin(
11730        VirtualBrowser_1,
11731        VirtualBrowser_1_Location,
11732        VirtualBrowser_1_Command,
11733        VirtualBrowser_1_Frame,
11734        document.URL,
11735        500,260,50,20,'#262',1.0);
11736    settleWin(
11737        VirtualBrowser_2,
11738        VirtualBrowser_2_Location,
11739        VirtualBrowser_2_Command,
11740        VirtualBrowser_2_Frame,
11741        'https://its-mce.jp/fa.jp/',
11742        500,280,150,100,'#444',1.0);
11743    settleWin(
11744        VirtualBrowser_3,
11745        VirtualBrowser_3_Location,
11746        VirtualBrowser_3_Command,
11747        VirtualBrowser_3_Frame,
11748        'http://gshell/gsh.go.html',
11749        'http://gshell.org/gshell/gsh.gsh.go.html',
11750        'https://goland.org',
11751        500,280,250,100,'#444',1.0);
11752        //1200,720,100,-50,'#444',0.125);
11753        //1200,720,100,-50,'#444',0.4);
11754    function WD_ClockUpdate(e){
11755        VirtualDesktop_1_Clock.innerHTML = DateShort();
11756        VirtualDesktop_1_Calender.innerHTML = DateHourMin();
11757    }
11758    window.setInterval(WD_ClockUpdate,500);
11759    window.setInterval(WD_EventSetup,500);
11760    WD_EventSetup1();
11761    WD_EventSetup2();
11762    WD_EventSetup3();
11763    WD_EventSetup4();
11764    WD_EventSetup5();
11765    WD_EventSetup6();
11766    WD_EventSetup7();
11767    WD_EventSetup8();
11768}
11769//VirtualDesktop_init();
11770Destroy_VirtualDesktop = function(){
11771    VirtualDesktop_1.style = "";
11772    VirtualBrowser_1.removeAttribute('style');
11773    VirtualBrowser_1.Location.innerHTML = '';
11774    VirtualBrowser_1_Frame.removeAttribute('src');
11775    VirtualBrowser_1_Frame.removeAttribute('style');
11776    VirtualBrowser_1_Frame.style="";
11777    VirtualBrowser_2.removeAttribute('style');
11778    VirtualBrowser_2_Location.innerHTML = '';
11779    VirtualBrowser_2_Frame.removeAttribute('src');
11780    VirtualBrowser_2_Frame.style="";
11781    VirtualBrowser_3.removeAttribute('style');
11782    VirtualBrowser_3_Location.innerHTML = '';
11783    VirtualBrowser_3_Frame.removeAttribute('src');
11784    VirtualBrowser_3_Frame.style="";
11785    VirtualBrowser_3_Frame.removeAttribute('style');
11786    VirtualBrowser_3_Location.innerHTML = '';
11787    VirtualBrowser_3_Frame.removeAttribute('src');
11788    VirtualBrowser_3_Frame.style="";
11789    VirtualBrowser_3_Frame.removeAttribute('style');
11790    VirtualBrowser_3_Frame.removeAttribute('style');
11791    GJfactory_1.style = "";
11792    iframe0.style = "";
11793    VirtualDesktop_1.style = "";
11794}
11795</script>
11796<!-- VirtualDesktop -->
11797<details>
11798</details>
11799</span>
11800<!-- ===== Work { ===== -->
11801<span id="SSSidebar_WorkCodeSpan">
11802</span>
11803</div>
11804<details><summary>SSSidebar</summary>
11805<div>
11806<h2>SSSidebar // 2020-0928 SatoxITS ( -->
11807<input id="SSSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11808<input id="SSSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11809<input id="SSSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11810<span id="SSSidebar_WorkScript"></span>
11811<input id="SSSidebar_WorkScript" type="button" value="WorkScript">
11812function SSSidebar_openWorkCodeView(){
11813    function SSSidebar_showWorkCode(){
11814        showHtmlCode(SSSidebar_WorkCodeView,SSSidebar_WorkCodeSpan);
11815    }
11816    SSSidebar_WorkCodeViewOpen.addEventListener('click',SSSidebar_showWorkCode);
11817}
11818SSSidebar_openWorkCodeView(); // should be invoked by an event
11819
11820console.log('-- SbSlider // 2020-1006-01 SatoxITS --');
11821function SbSlider(id){
11822    sidebar = document.getElementById('secondary');
11823    console.log('primary+' + 'primary' + 'secondary'+ 'secondary'+ 'main'+ 'main'+ ' ');
11824    wrap = sidebar.parentNode;
11825    console.log('-- SbSlider parent is '+wrap+', #' +wrap.id+ ' .' +wrap.class);
11826

```

```

11826//wrap = wrap.parentNode;
11827//console.log('-- SbSlider parent is '+wrap+', #' +wrap.id+' .' +wrap.className);
11828//wrap.appendChild(sidebar);
11829//console.log('-- SbSlider parent is '+wrap+', #' +wrap.id+' .' +wrap.className);
11830//nsb = sidebar.cloneNode();
11831    nsb = sidebar;
11832    nsb.style.width = '100%';
11833    nsb.innerHTML = document.createElement('div');
11834    slider = 'SbSlider';
11835    slider.appendChild(nsb);
11836    slider.setAttribute('class','SbSlider');
11837    slider.style.position = 'relative';
11838    slider.style.position = 'fixed';
11839    slider.style.display = 'block'; // 'inline';
11840    slider.style.zIndex = 10000;
11841    // nsb.style.zIndex = 20000;
11842    nsb.style.position = 'absolute';
11843    nsb.style.minWidth = '80px';
11844    nsb.style.left = '0px';
11845    nsb.style.top = '0px';
11846
11847    w = window.innerWidth;
11848    console.log('SliderWidth '+w+: ',(w/3)+'px');
11849    if( w < 640 ){
11850        slider.style.setProperty('width',(w/3) + 'px','important');
11851    }
11852    main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
11853
11854    slider.style.resize = 'both';
11855    slider.draggable = 'true';
11856    wrap.appendChild(slider);
11857    console.log('-- added SbSlider');
11858    //nsb.addEventListener('scroll',SbScrolled);
11859
11860    button = document.createElement('div');
11861    buttons.id = 'NaviButtons';
11862    buttons.setAttribute('class','NaviButtons');
11863    buttons.style.position = 'absolute';
11864    buttons.innerHTML += '<'+p+><a href="#PopOffPost" draggable="true">TOP<'+a+><'+p+>';
11865    buttons.innerHTML += '<'+p+><a href="#EndOffPost" draggable="true">END<'+p+>';
11866    page.appendChild(button);
11867    buttons.style.position = 'fixed';
11868    buttons.style.width = '100px';
11869    buttons.style.height = '18px';
11870    buttons.style.top = '32px';
11871    buttons.style.left = parseInt(w) * 1.0 + 'px';
11872    console.log('-- SbSlider installed (~)/ SatoxITS');
11873}
11874//window.addEventListener('load',SetSideBar); // after load
11875DestroyNaviButtons = function(){
11876    nb = document.getElementById('NaviButtons');
11877    if( nb != null ){
11878        nb.parentNode.removeChild(NaviButtons);
11879    }
11880}
11881</script>
11882
11883// 2020-10-06 its-more.jp-blog-60000-style.css
11884<-- {
11885    <style>
11886        #naviButtons {
11887            position:fixed;
11888            display:block;
11889            width:100%;
11900            xtop:100px;
11901            xleft:0px;
11902            z-index:30000;
11903            font-size:10pt;
11904            color:#fff !important;
11905            text-align:center;
11906            background-color:rgba(230,230,230,0.01);
11907        }
11908        #naviButtons a {
11909            color:#2a2 !important;
11910            font-size:20px;
11911            text-align:center;
11912            xtext-shadow:2px 2px #fff;
11913            xsize:1em;
11914            padding:6px;
11915            margin:10px;
11916            border:1px solid #288 !important;
11917            border-radius:3px;
11918            background-color:rgba(160,160,160,0.05);
11919        }
11920        #SbSlider {
11921            overflow:auto;
11922            resize:both !important;
11923            xoverflow-y:hidden !important;
11924            height:100px !important;
11925            xfont-size:line !important;
11926            position:fixed !important;
11927            left:0px;
11928            top:0px;
11929            xxwidth:180px;
11930            width:180px;
11931            min-width:80px;
11932            height:100% !important;
11933            background-color:rgba(100,100,200,0.1);
11934        }
11935        #secondary {
11936            position:fixed;
11937            left:0px;
11938            top:0px;
11939            xxxz-index:60000;
11940            scroll-behavior: overflow !important;
11941            xbackground:xxxxxxxxxxxxxxxxxxxxxxwidth:18% !important;
11942            padding-left:4px;
11943            color:#fff;
11944            font-size:0.5em;
11945            background-color:rgba(64,160,64,0.6) !important;
11946            white-space:nowrap;
11947        }
11948        #secondary a {
11949            color:#fff !important;
11950            text-decoration:disabled !important;
11951        }
11952        #primary {
11953            position:relative;
11954            width:75% !important;
11955            left:25% !important;
11956        }
11957        #main {
11958            position:relative;
11959            width:75% !important;
11960            left:25% !important;
11961        }
11962        #nowTime {
11963            color:#a0ffa0;
11964            font-size:16pt !important;
11965            xfont-size:108 !important;
11966            font-weight:bold;
11967            text-shadow:1px 1px #fff;
11968        }
11969        .navigation-top {
11970            color:#22a !important;
11971            border:0px;
11972            background-color:rgba(220,220,220,0.1);
11973        }
11974
11975        .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
11976            display: block;
11977            xxwidth: 1em;
11978            xoverflow: auto;
11979            xxheight: 1em;
11980        }
11981        .invisible-scrollbar ::-webkit-scrollbar {
11982            xdisplay: none;
11983            width:1px !important;
11984            height:1px !important;
11985        }
11986        .mostly-customized-scrollbar ::-webkit-scrollbar {
11987            width: 2px;

```

```
11988 height: 2px;
11989 xbackground-color: #aaa; xxx:or add it to the track;
11990 }
11991 .mostly-customized-scrollbar ::-webkit-scrollbar-thumb {
11992 background: #000;
11993 }
11994 </style>
11995 -->
11996
11997
11998 <!-- SBSidebar_WorkCodeSpan -->
11999 <-- SBSidebar_WorkCodeSpan -->
12000 <-- SBSidebar_WorkCodeSpan -->
12001 //<-- ===== Work ===== -->
12002
12003
12004 //<-- ===== Work ===== -->
12005 <span id="Affiliate_WorkCodeSpan">
12006 /*
12007 details id="Affiliate_Test"><summary>Affiliate</summary>
12008 <!-- Affiliate // 2020-1010 SatoshiTS -->
12009 <div id="AffViewDock">
12010 <div id="AffView" class="AffView" draggable="true" style="">
12011 <div id="AffSet" class="AffPlate">
12012 <iframe id="aff_0" class="AffItem"></iframe>
12013 <iframe id="aff_1" class="AffItem"></iframe>
12014 <iframe id="aff_2" class="AffItem"></iframe>
12015 <iframe id="aff_3" class="AffItem"></iframe>
12016 <iframe id="aff_4" class="AffItem"></iframe>
12017 <iframe id="aff_5" class="AffItem"></iframe>
12018 </div>
12019 </div>
12020 <h2>Supportive Affiliate</h2>
12021 <style>
12022 .AffView {
12023 z-index:0;
12024 position:-xscroll;
12025 overflow-y:scroll;
12026 position:fixed;
12027 max-width:2560px;
12028 max-height:100%;
12029 width:270px;
12030 left:75px;
12031 height:95%;
12032 resize:both;
12033 left:-10px;
12034 margin-top:40px;
12035 xleft:0;
12036 xright:0;
12037 display:block;
12038 border:4px inset rgba(255,255,255,0.1);
12039 background-color:rgba(255,255,255,0.1);
12040 }
12041 .AffView:hover {
12042 z-index:1;
12043 width:300px;
12044 overflow:scroll;
12045 border:4px inset #ffc;
12046 background-color:rgba(80,80,255,0.2);
12047 background-color:#ffc;
12048 }
12049 .AffPlate:hover {
12050 border-left:4px dashed #888;
12051 }
12052 .AffPlate {
12053 overflow-x:visible;
12054 border-left:4px dashed rgba(255,255,255,0.1);
12055 max-width:2560px;
12056 max-height:2880px;
12057 margin-top:10px;
12058 margin-bottom:10px;
12059 margin-left:4px;
12060 width:30px;
12061 xheight:1440px;
12062 }
12063 .AffItem {
12064 overflow-x:visible;
12065 overflow-y:scroll;
12066 max-width:2560px;
12067 max-height:1440px;
12068 z-index:0;
12069 display:block;
12070 xposition:fixed;
12071 xtop:0;
12072 position:relative;
12073 /*/left:300px;
12074 xresize:both;
12075 padding:0px;
12076 width:60px;
12077 height:40px;
12078 max-height:800px !important;
12079 margin-top:0px;
12080 margin-left:0px;
12081 margin-right:0% !important;
12082 border:4px inset #ccc;
12083 transform:scale(0.5);
12084 background-color:rgba(255,255,255,0.2);
12085 xxalign:right;
12086 }
12087 .AffItem:hover {
12088 border:16px inset #bbf;
12089 }
12090 </style>
12091 <input id="Affiliate_WorkViewOpen" type="button" value="ShowCode">
12092 <input id="Affiliate_WorkOpenSnapshot" type="button" value="Snapshot">
12093 <input id="Affiliate_WorkCodeSignature" type="button" value="Signature">
12094 <span id="Affiliate_WorkScript"></span>
12095 <script id="Affiliate_WorkScript">
12096 function Affiliate_OpenWorkCodeView(){
12097   function Affiliate_ShowWorkCode(){
12098     showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
12099   }
12100   Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_ShowWorkCode);
12101 }
12102 Affiliate_OpenWorkCodeView(); //// should be invoked by an event
12103
12104 //<iframe id="aff_0" xsrc="https://stackoverflow.com/tags" class="AffItem"></iframe>
12105 //<iframe id="aff_1" xsrc="https://developer.mozilla.org/en-US/docs/Web" class="AffItem"></iframe>
12106 wasAffIsSetup = false;
12107 Affiliate_Test.addEventListener('click',Aff_Setup);
12108 function Aff_Setup(){
12109   if(Aff_isSetup){return;} Aff_isSetup = true;
12110   parent = document.documentElement;
12111   parent.appendChild(idAffView);
12112   AffView.style.top = '0px';
12113   AffView.style.left = (window.innerWidth - 280) + 'px';
12114
12115   var off = 100;
12116   zoom = 0.5;
12117   zoom += 0.3;
12118   lefttx = window.innerWidth - 300;
12119   left = lefttx + 'px';
12120   left = '-130 + ' + px;
12121   console.log('aff-init window.innerWidth=' + window.innerWidth);
12122   w = 1000;
12123   h = 560;
12124
12125   aff_0.src = "../gshell/gsh.go.html";
12126   aff_0.src = "https://golang.org";
12127   aff_2.src = "https://drafts.csawg.org/";
12128   aff_3.src = "https://spec.whatwg.org/dev/";
12129   aff_5.src = "https://wikipedia.org";
12130   aff_5.src = "https://bing.com/translator";
12131
12132   /parent.appendChild(aff_0);
12133   aff_0.style.width = zoom+w+px;
12134   aff_0.style.height = zoom+h+px;
12135   aff_0.style.left = left;
12136   /aff_0.style.top = off+px; off += ozoom*h;
12137   aff_0.draggable = 'true';
12138
12139   /parent.appendChild(aff_1);
12140   aff_1.style.width = zoom+w+px;
12141   aff_1.style.height = zoom+h+px;
12142   aff_1.style.left = left;
12143   /aff_1.style.top = off+px; off += ozoom*h;
12144   aff_1.style.top = '-150px';
12145   aff_1.draggable = 'true';
12146
12147   /parent.appendChild(aff_2);
12148   aff_2.style.width = zoom+w+px;
12149   aff_2.style.height = zoom+h+px';
```

```

12150 aff_2.style.left = left;
12151 //aff_2.style.top = off+'px'; off += ozoom*h;
12152 aff_2.style.top = '-300px';
12153 aff_2.draggable = 'true';
12154
12155 //parent.appendChild(aff_3);
12156 aff_3.style.transform = 'scale(0.25)';
12157 aff_3.style.width = '2zoom*px';
12158 aff_3.style.height = '2zoom*px';
12159 aff_3.style.border = '32px inset #ccc';
12160 //aff_3.style.left = '-390 +' + px; //left*2;
12161 //aff_3.style.left = (leftx - 265) + 'px';
12162 aff_3.style.left = '-395 +' + px';
12163 //aff_3.style.top = '-355+off)' + px'; off += ozoom*h;
12164 aff_3.style.top = '-600px';
12165 aff_3.draggable = 'true';
12166
12167 //parent.appendChild(aff_4);
12168 aff_4.style.width = 'zoom*px';
12169 aff_4.style.height = 'zoom*h' + px';
12170 aff_4.style.left = leftx + px;
12171 //aff_4.style.top = off+'px'; off += ozoom*h;
12172 aff_4.style.top = '-900px';
12173 aff_4.draggable = 'true';
12174
12175 //parent.appendChild(aff_5);
12176 aff_5.style.transform = 'scale(0.300)';
12177 aff_5.style.width = 'zoom*(w*1.67)' + px';
12178 aff_5.style.height = 'zoom*(h*1.67)' + px';
12179 aff_5.style.border = '25px inset #ccc';
12180 aff_5.style.left = '-308' + px';
12181 aff_5.style.left = (leftx + px) + px';
12182 //aff_5.style.top = ('-95+off)' + px'; off += ozoom*h;
12183 aff_5.style.left = '0px';
12184 aff_5.style.top = '0px';
12185 aff_5.style.top = '-1150px';
12186 //aff_5.style.align = 'right';
12187 aff_5.draggable = 'true';
12188
12189 window.addEventListener('resize',affresize);
12190 }
12191 function affresize(){
12192   AffView.style.left = (window.innerWidth - 280) + 'px';
12193   AffView.style.width = window.innerWidth - 400;
12194   left = leftx + 'px';
12195   console.log('aff-resize window.innerWidth=' + window.innerWidth);
12196 }
12197 //window.addEventListener('resize',affresize);
12198 //document.addEventListener('resize',affresize);
12199 //gsh.addEventListener('resize',affresize);
12200
12201 function ResetAffView(){
12202   AffViewDock.appendChild(AffView);
12203   AffView.removeAttribute('style');
12204   aff_0.removeAttribute('style');
12205   aff_0.removeAttribute('src');
12206   aff_1.removeAttribute('src');
12207   aff_1.removeAttribute('style');
12208   aff_1.draggable = 'true';
12209   aff_2.removeAttribute('style');
12210   aff_2.draggable = 'true';
12211   aff_3.removeAttribute('style');
12212   aff_3.draggable = 'true';
12213   aff_4.removeAttribute('src');
12214   aff_4.removeAttribute('style');
12215   aff_5.removeAttribute('src');
12216   aff_5.removeAttribute('style');
12217 }
12218 </script>
12219 <details>
12220 <!-- Affiliate_WorkCodeSpan } -->
12221 <!-- //<span>
12222 //<!-- ===== Work } ===== -->
12223
12224
12225 <!-- ===== Work { ===== -->
12226 <span id="TextCanvas_WorkCodeSpan">
12227
12228 <details id="TextCanvas_Section"><summary id="TextCanvas_Summary">TextCanvas</summary>
12229 <!-- ----- TextCanvas // 2020-1013 SatoxITS { -->
12230
12231 <details id="FontSelect_Section"><summary id="FontSelect_Summary">Font Selection</summary>
12232 <h2>Font Selection</h2>
12233 <div id="FontList"></div>
12234 <div id="FontList"></div>
12235 <style>
12236 #FontList {
12237   overflow:visible;
12238   background-color:rgba(240,245,255,1.0) !important;
12239 }
12240 #FontList td {
12241   font-size:16px;
12242   padding:0px;
12243   padding-left:2px;
12244   padding-right:1px;
12245   margin:0px;
12246   line-height:1.2;
12247   border:0px;
12248 }
12249 #FontList tdi:hover {
12250   background-color:#222;
12251 }
12252 #FontList tr:hover {
12253   color:#fff;
12254   background-color:#000;
12255   xborder:1px solid #000;
12256 }
12257 .xcourier { colr:#000; font-size:16px; font-family:courier; }
12258 .xcursive { colr:#000; font-size:16px; font-family:cursive; }
12259 .xfantasy { colr:#000; font-size:16px; font-family:fantasy; }
12260 .xgeorgia { colr:#000; font-size:16px; font-family:georgia; }
12261 .xmonospace { colr:#000; font-size:16px; font-family:monospace; }
12262 </style>
12263 <script>
12264 function fontstr(name,text){
12265   /tr + '<'+tr style="font-family:' + name + "'>\n";
12266   tr + '<td style="font-family:' + name + "'>\n";
12267   tr + '<td style="font-family:Arial;font-size:12pt;">' + name + '</' + td + '>';
12268   tr + '<td data-fsty="n">' + text + '</' + td + '>';
12269   tr + '<td data-fsty="b">' + text + '</' + td + '>';
12270   tr + '<td data-fsty="i">' + text + '</' + td + '>';
12271   tr + '<td data-fsty="bi">' + text + '</' + td + '>';
12272   tr + '<td data-fsty="bi">' + text + '</' + td + '>';
12273   tr + '<td data-fsty="i">' + text + '</' + td + '>';
12274   tr + '<td data-fsty="bi">' + text + '</' + td + '>';
12275   return tr;
12276 }
12277 function lsfont(){
12278   text = 'GShell-Go012';
12279
12280   f1 = '';
12281   f1 += '<table>\n';
12282   f1 += fontstr('Arial',text);
12283   f1 += fontstr('Courier',text);
12284   f1 += fontstr('Courier New',text);
12285   f1 += fontstr('Georgia',text);
12286   f1 += fontstr('Inkfree',text);
12287   f1 += fontstr('Verdana',text);
12288   f1 += fontstr('Times',text);
12289   f1 += fontstr('Osaka',text);
12290   f1 += fontstr('Meiryo',text);
12291   f1 += fontstr('YuMincho',text);
12292
12293 //f1 += fontstr('Roman',text);
12294 //document.fonts.load("30px cursive");
12295 f1 += fontstr('Seri',text);
12296 f1 += fontstr('Consolas',text);
12297 f1 += fontstr('System-UT',text);
12298 f1 += fontstr('Monospace',text);
12299 f1 += fontstr('Cursive',text);
12300 f1 += fontstr('Fantasy',text);
12301 f1 += '</table>\n';
12302
12303 if( false ){
12304   fss = document.fonts.entries(); // FontFaceSet
12305   console.log('FSS=' + fss);
12306   while( true ){
12307     font = fss.next();
12308     if( font.done ){
12309       break;
12310     }
12311     f1 += font.value[0] + '<br>';

```

```

12312     }
12313   }
12314   FontList.innerHTML = fl;
12315 }
12316 function selectFont(e){
12317   t = e.target;
12318   let fsty = '';
12319   for(i = 0; i < 4; i++ ){
12320     //console.log('FontSelect '+t.nodeName+' #' +t.id+ ' ' +t.style);
12321     if( t.nodeName == 'TD' ){
12322       //console.log('FontSelect '+t.outerHTML);
12323       if( t.hasAttribute('data-fsty') ){
12324         fsty = t.getAttribute('data-fsty');
12325       //console.log('FontSelect = ' + fsty);
12326     }
12327   }
12328   if( t.nodeName != 'TD' ){
12329     if( t.style != '' ){
12330       if( t.style.fontFamily != '' ){
12331         break;
12332       }
12333     }
12334   }
12335   t = t.parentNode;
12336 }
12337 if( t.style != '' ){
12338   font = t.style.fontFamily;
12339   //console.log('FontSelect: '+font);
12340   //console.log('FontSelect == ' + fsty);
12341   if( font != '' ){
12342     sel = document.getElementById("TextCanvas_1_Font");
12343     if( sel != null ){
12344       if( fsty != '' ){
12345         TextCanvas_1_Bold.checked = 0 <= fsty.indexOf('b');
12346         TextCanvas_1_Italic.checked = 0 <= fsty.indexOf('i');
12347       }
12348       sel.value = font;
12349       RedrawTextCanvas();
12350     }else{
12351       alert('Event: ' + e.target.nodeName + ' #' + font);
12352     }
12353   }
12354 }
12355 FontList.addEventListener('click',selectFont);
12356 document.fonts.onloadingdone = function(fsse){
12357   //alert('font-loaded '+fsse.fontfaces.length);
12358 }
12359 }
12360 function Fontlist_Setup(){
12361   if( FontSelect_Summary.open ){
12362     lsfont();
12363   }
12364 }
12365 FontSelect_Summary.addEventListener('click',lsfont);
12366 </script>
12367 </details>
12368
12369 <h2>Drawing Text on Canvas</h2>
12370 <div id="2020-10-12_Drawing_text_on_Canvas" style="border:1px solid black; padding:10px; margin-bottom:10px;>
12371 <div id="TextCanvas_1_Label" class="CanvasLabel">
12372   <input id="TextCanvas_1_Clear" type="button" value="Clear">
12373   <input id="TextCanvas_1_Draw" type="button" value="Draw">
12374   <input id="TextCanvas_1_Font" class="CanvasPanel" type="text" size="14" value="Georgia">
12375   <input id="TextCanvas_1_Bold" class="CanvasBox" type="checkbox" checked="">>Bold
12376   <input id="TextCanvas_1_Italic" class="CanvasBox" type="checkbox" checked="">>Italic
12377   <input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
12378   <input id="TextCanvas_1_Color" class="CanvasPanel" type="text" size="6" value="#22a">
12379 <!-- to be PBlue series ? -->
12380 <p>
12381   <input id="TextCanvas_1_Text" class="TextCanvasText" type="text" size="50" value="GShell">
12382 </p>
12383 <div id="TextCanvas_1_BgImage" style="background-color:#fff; position:relative; height:100px; width:100px; border:1px solid black; margin-bottom:10px; overflow:hidden; border-radius:10px; border:none; background-size:cover; background-position:center; background-repeat:no-repeat; z-index:-1; opacity:0.8; filter:alpha(opacity=80);>
12384 <img id="TextCanvas_1_BgImage" class="CanvasInImage" style="position: absolute; left: 0; top: 0; width: 100%; height: 100%; z-index: -1; border: none; border-radius: 10px; background-color: #fff; background-size: cover; background-position: center; background-repeat: no-repeat; opacity: 1; filter: alpha(opacity=1);>
12385 </div>
12386 <div id="TextCanvas_1" class="TextCanvas" style="border:1px solid black; width:100px; height:100px; position: relative; border-radius:10px; border:none; background-color:#fff; margin-bottom:10px; z-index: 0; opacity: 1; filter: alpha(opacity=1);>
12387 <div id="TextCanvas_1_Label" style="position: absolute; left: 0; top: 0; width: 100%; height: 100%; z-index: -1; border: none; border-radius: 10px; background-color: #fff; background-size: cover; background-position: center; background-repeat: no-repeat; opacity: 1; filter: alpha(opacity=1);>
12388 <div id="TextCanvas_1_ToImage" class="PanelButton" type="button" value="ToImage">
12389 <input id="TextCanvas_1_TOPNG" class="PanelRadio" type="radio" name="ImageType" value="ToPNG" checked="">>PNG
12390 <input id="TextCanvas_1_TOJPEG" class="PanelRadio" type="radio" name="ImageType" value="ToJPEG">>JPEG
12391 <input id="TextCanvas_1_DataURL" class="PanelBox" type="checkbox" checked="">>DataURL
12392 <div class="CanvasInImage"><br><img id="TextCanvas_1_Image" class="CanvasImage" src=""></div>
12393 <div id="TextCanvas_1_BgImage" class="CanvasInImage" style="background-color:#fff; position: absolute; left: 0; top: 0; width: 100%; height: 100%; z-index: -1; border: none; border-radius: 10px; background-size: cover; background-position: center; background-repeat: no-repeat; opacity: 1; filter: alpha(opacity=1);>
12394 <div id="TextCanvas_1_DataUrlView" class="DataUrlView" style="border:1px solid black; width:100px; height:100px; position: absolute; left: 0; top: 0; z-index: 1; border-radius: 10px; border:none; background-color:#fff; background-size: cover; background-position: center; background-repeat: no-repeat; opacity: 1; filter: alpha(opacity=1);>
12395 <span id="TextCanvas_1_DataUrlText" style="border:1px solid black; width:100px; height:100px; position: absolute; left: 0; top: 0; z-index: 1; border-radius: 10px; border:none; background-color:#fff; background-size: cover; background-position: center; background-repeat: no-repeat; opacity: 1; filter: alpha(opacity=1);>
12396 </div>
12397 </div>
12398 <div id="TextCanvas_1_CommandUsageText" style="border:1px solid black; width:100px; height:100px; position: absolute; left: 0; top: 0; z-index: 1; border-radius: 10px; border:none; background-color:#fff; background-size: cover; background-position: center; background-repeat: no-repeat; opacity: 1; filter: alpha(opacity=1);>
12399 <style>
12400 .CommandUsageText {
12401   font-family:Courier New;
12402 }
12403 .TextCanvas {
12404   border:1px solid #000;
12405   resize:both;
12406   display:inline !important;
12407 }
12408 .DataUrlView {
12409   width:100px;
12410   font-size:10pt;
12411   font-family:Courier New, monospace;
12412   color:#000;
12413   background-color:#eee;
12414   margin-bottom:10px;
12415   xdisplay:block;
12416   xoverflow:scroll;
12417 }
12418 .CanvasImage {
12419   border:1px dashed #000;
12420 }
12421 .TextCanvasText {
12422   font-size:12pt;
12423   width:100px;
12424 }
12425 .CanvasLabel {
12426   font-size:10pt;
12427   color:#000;
12428 }
12429 .CanvasInImage {
12430   width:100px;
12431   height:100px;
12432   margin-bottom:10px;
12433   color:rgba(32,160,32,0.5);
12434   text-shadow:3px 3px #eee;
12435   background-color:#eee;
12436   xborder:1px solid #000;
12437   font-size:18pt;
12438   vertical-align:middle;
12439 }
12440 .PanelRadio {
12441   font-size:12pt !important;
12442   color:#000 !important;
12443   vertical-align:middle;
12444 }
12445 .CanvasBox {
12446   vertical-align:middle;
12447   margin-left:4px !important;
12448   margin-right:2px !important;
12449 }
12450 .CanvasPanel {
12451   vertical-align:middle !important;
12452   height:14px !important;
12453   width:14px !important;
12454   padding:2px !important;
12455   margin:4px !important;
12456   margin-left:4px !important;
12457   margin-right:2px !important;
12458   font-size:10pt !important;
12459   font-family:Georgia !important;
12460   color:#000;
12461   display:inline !important;
12462 }
12463 .TextCanvasPanel {
12464   vertical-align:middle;
12465   font-size:10pt !important;
12466   font-family:Georgia !important;
12467   color:#000;
12468   display:inline !important;
12469 }
12470 .Panelbutton {
12471   font-size:10pt !important;
12472   font-family:Georgia !important;
12473   vertical-align:middle;

```

```

12474 width:14px !important;
12475 height:14px !important;
12476 line-height:1.2 !important;
12477 padding:2px !important;
12478 margin:1px !important;
12479 display:inline-block !important;
12480 padding:1px !important;
12481 color:#fff !important;
12482 background-color:#228 !important;
12483 }
12484 
```

```

12636 height:400px;
12637 }
12638 .ShadingHTML {
12639 z-index:2;
12640 position:relative;
12641 display:block;
12642 top:0px;
12643 left:0px;
12644 overflow:scroll;
12645 width:100px;
12646 font-size:12pt;
12647 font-family:Courier New;
12648 color:#666;
12649 background-color:olivedrab;
12650 background-rgba(200,255,200,0.4);
12651 height:400px;
12652 }
12653 .ShadingCanvas {
12654 z-index:3;
12655 position:relative;
12656 display:block;
12657 top:0px;
12658 left:100px;
12659 resize:both;
12660 border:1px solid #000;
12661 }
12662 </style>
12663 <input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12664 <input id="Shading_OpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12665 <input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12666 <span id="Shading_WorkCodeView"></span>
12667 <script id="sh" type="text/javascript">
12668 function Shading_OpenWorkCodeView(){
12669   function Shading_ShowWorkCode(){
12670     showHtmlCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
12671   }
12672   Shading_WorkCodeViewOpen.addEventListener('click',Shading_ShowWorkCode);
12673 }
12674 const BR = '<'+br+'';
12675 Shading_OpenWorkCodeView(); // should be invoked by an event
12676 function sh_onClick(e){
12677   Shading_1_Log.innerHTML += 'Click '+e.target.nodeName+'#'+e.target.id
12678   +' offsetX:'+e.offsetX+', '+e.offsetY+','
12679   +' clientX:' + e.clientX +
12680   +' pageX:' + e.pageX +
12681   +' screenX:' + e.screenX +
12682   +' +BR;
12683   e.stopPropagation();
12684   e.preventDefault();
12685 }
12686 function sh_onKeyUp(e){
12687   if( Shading_1.style.zIndex == '' ){
12688     Shading_1.style.zIndex = 0;
12689   }
12690   zi = parseInt(Shading_1.style.zIndex);
12691   if( e.key.length == 1 ){
12692     Shading_1_Html.innerHTML += e.key;
12693   }
12694   if( e.key == '0' ) { zi = 0; }else
12695   if( e.key == '+' ) { zi += 1; }else
12696   if( e.key == '-' ) { zi -= 1; }else
12697   if( e.key == 'c' ) {
12698     Shading_1_Log.innerHTML = '';
12699   }
12700   else{
12701     if( e.key == 'r' ){
12702       Shading_1.style.position = "relative";
12703       Shading_1.style.top = '0px';
12704       Shading_1.style.left = '0px';
12705       zi = 0;
12706     }
12707     if( e.key == 'j' || e.code == 'ArrowDown' ){
12708       topx = parseInt(Shading_1.style.top) + 50;
12709       Shading_1.style.top = topx + 'px';
12710     }
12711     if( e.key == 'k' || e.code == 'ArrowUp' ){
12712       topx = parseInt(Shading_1.style.top) - 50;
12713       Shading_1.style.top = topx + 'px';
12714     }
12715     if( e.key == 'l' || e.code == 'ArrowRight' ){
12716       lefty = parseInt(shading_1.style.left) + 50;
12717       Shading_1.style.left = lefty + 'px';
12718     }
12719     if( e.key == 'h' || e.code == 'ArrowLeft' ){
12720       lefty = parseInt(shading_1.style.left) - 50;
12721       Shading_1.style.left = lefty + 'px';
12722     }
12723     else{
12724       if( e.key == 'a' ){
12725         Shading_1.style.position = "absolute";
12726         Shading_1.style.top = '0px';
12727         Shading_1.style.left = '0px';
12728       }
12729     }
12730     Shading_1.style.zIndex = zi;
12731     Shading_1_Log.innerHTML += 'Keyup.'+e.target.nodeName+'#'+e.target.id
12732     +'Up('+e.key+''+e.code+')';
12733     +'z-index:' +zi+''+shading_1.style.zIndex
12734     +'top:' +shading_1.style.top
12735     +' +BR;
12736   e.stopPropagation();
12737   e.preventDefault();
12738 }
12739 function sh_onKeyDown(e){
12740   Shading_1_Log.innerHTML += 'Keypad.'+e.target.nodeName+'#'+e.target.id
12741   +'Down('+e.key+''+e.code+')';
12742   e.stopPropagation();
12743   e.preventDefault();
12744 }
12745 function Shading_Setup(){
12746   Shading_1_Log.innerHTML += '<'+h4+>Click here</'+h4+>;
12747   Shading_1.addEventListener('keydown',sh_onkeydown);
12748   Shading_1.addEventListener('keyup',sh_onkeyup);
12749   Shading_1.addEventListener('click',sh_onclick);
12750   Shading_1.addEventListener('click',sh_onclick);
12751   Shading_1.addEventListener('click',sh_onclick);
12752   Shading_1.Log.style.top = "-400px";
12753   Shading_1.Log.style.left = "-200px";
12754   Shading_1.appendChild(Shading_1_Canvas);
12755   Shading_1_Canvas.style.width = "300px";
12756   Shading_1_Canvas.style.height = "300px";
12757   Shading_1_Canvas.style.position = "relative";
12758   Shading_1_Canvas.style.top = "-750px";
12759   Shading_1_Canvas.style.left = "100px";
12760
12761 const ctx = Shading_1_Canvas.getContext('2d');
12762 ctx.fillStyle = "rgba(160,0,0,0.9)";
12763 ctx.fillRect(50,50,40,40);
12764 ctx.fillStyle = "rgba(0,160,0,0.9)";
12765 ctx.fillRect(60,60,40,40);
12766 ctx.fillStyle = "rgba(0,0,160,0.9)";
12767 ctx.fillRect(70,70,40,40);
12768
12769 function Reset_ShadingCanvas(){
12770   Shading_1_Log.removeAttribute('style');
12771   Shading_1_Log.innerHTML = '';
12772   Shading_1_Canvas.style = "";
12773   //Shading_1_Canvas.removeAttribute('style');
12774 }
12775
12776</script>
12777<details>
12778<!-- Shading_WorkCodeSpan -->
12779<!-- Shading_WorkCodeSpan -->
12780<!-- /<span>
12781//<-- ===== Work ===== -->
12782<!-- ===== Work ===== -->
12783<!-- ===== Work ===== -->
12784<!-- ===== Work ===== -->
12785<span id="Charmap_WorkCodeSpan">
12786<!--
12787<details id="Charmap" summary="Character Map Mandala">
12788<!-- UnicodeCharMap // 2020-1015 SatoyITS -->
12789<h2>Unicode Character Map</h2>
12790<note>Code 0x0000 - 0xFFFF / 16px / 3200 x 3200 px / zoom:0.25</note>
12791<div id="Charmap_1_Frame">
12792<div id="Charmap_1_Text" class="Charmap">
12793</div>
12794</div>
12795<br>
12796
12797<style>

```

```
12799 //Charmap_1_Frame {
12800   overflow:scroll;
12801   height:3200px; width:3200px;
12802   xtransform:scale(0.5);
12803   zoom:0.25;
12804   resize:both;
12805   background-color:#fff;
12806 }
12807 .Charmap_1 {
12808   xzoom:0.25;
12809   font-size:16px;
12810   line-height:1.0;
12811   xxfont-family:Georgia;
12812   color:#000;
12813 }
12814 <style>
12815 </style>
12816 function charmapgen(){
12817   text = '';
12818   for cc = 0; cc < 0x10000; cc++ ){
12819     text += String.fromCharCode(cc);
12820   }
12821   Charmap_1.Text.innerHTML = text;
12822 }
12823 Charmap_Work.addEventListener('click',charmapgen);
12824 //charmapgen();
12825 </script>
12826 <input id="Charmap_WorkCodeViewOpen" class="HmlCodeViewButton" type="button" value="ShowCode">
12827 <input id="Charmap_WorkOpenSnapshot" class="HmlCodeViewButton" type="button" value="Snapshot">
12828 <input id="Charmap_WorkCodeSignature" class="HmlCodeViewButton" type="button" value="Signature">
12829 <span id="Charmap_WorkSpanView"></span>
12830 <script id="Charmap_WorkSpanView">
12831   function Charmap_openWorkCodeView(){
12832     function Charmap_showWorkCode(){
12833       showHmlCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
12834     }
12835     Charmap_WorkCodeViewOpen.addEventListener('click',Charmap_showWorkCode);
12836   }
12837   Charmap_openWorkCodeView(); // should be invoked by an event
12838 </script>
12839 <details>
12840 <!-- Charmap_WorkCodeSpan -->
12841 </--> </span>
12842 <!-- ===== Work -->
12843 </div>
12844 <!-- ===== Work { ===== -->
12845 <!--> <span id="Pointillism_WorkCodeSpan">
12846 </span>
12847 </div>
12848 <div details=><summary>Collaborated Pointillism</summary>
12849 <!--> <span id="Pointillism_Collaborated" class="Pointillism"><-->
12850 <h2><a name="Pointillism" class="Pointillism" href="#Pointillism">collaborated Pointillism</a></h2>
12851 <input id="Pointillism_1_Share" type="checkbox" class="HmlCodeViewButton" value="Share"> Share
12852 <input type="button" class="HmlCodeViewButton" value="Pointillism_1_ResetCanvas()" onclick="Pointillism_1_ResetCanvas()"/>
12853 <input type="button" class="HmlCodeViewButton" value="Pointillism_1_ClearCanvas()" onclick="Pointillism_1_ClearCanvas()"/>
12854 <input type="button" class="HmlCodeViewButton" value="Clear" onclick="Pointillism_1_ClearCanvas()"/>
12855 <input type="button" class="HmlCodeViewButton" onclick="Pointillism_1_ReplayCanvas()" value="Replay">
12856 <input type="button" class="HmlCodeViewButton" onclick="Pointillism_1_RepeatCanvas()" value="Repeat">
12857 <input type="button" class="HmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Save">
12858 <input type="button" class="HmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Load">
12859 <div id="Pointillism_1" class="Pointillism">
12860   <div id="Pointillism_1_Share" class="Pointillism_Share">
12861     <span id="Pointillism_1_Unit_1" class="Pointillism_Unit">
12862       <span id="Pointillism_1_XY_1" class="Pointillism_XY"><--> XY1</span>
12863       <span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote"><--> XY1-remote</span>
12864     <canvas id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
12865   </span>
12866   <div id="Pointillism_1_Unit_2" class="Pointillism_Unit">
12867     <span id="Pointillism_1_XY_2" class="Pointillism_XY"><--> XY2</span>
12868     <span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote"><--> XY2-remote</span>
12869   <div id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" width="300px" height="300px"></div>
12870   </div>
12871 </div>
12872 </div>
12873 </div>
12874 </div>
12875 <style>
12876 .Pointillism {
12877   display:block;
12878   resize:both;
12879   width:60px;
12880   height:380px;
12881   margin:10px;
12882   min-width:470px;
12883   min-height:470px;
12884   background-color:#eee;
12885   overflow:scroll;
12886   font-size:16px;
12887   font-family:Georgia;
12888   color:#000;
12889   vertical-align:middle;
12890 }
12891 .Pointillism_Unit {
12892   position:relative;
12893   top:0px;
12894   display:block;
12895   overflow:scroll;
12896   width:300px;
12897   height:350px;
12898   margin:5px;
12899   padding:10px;
12900   background-color:rgba(255,255,127,0.7);
12901 }
12902 .Pointillism_XY {
12903   display:block;
12904   vertical-align:middle;
12905   width:90px;
12906   margin:5px;
12907   font-size:12px;
12908   line-height:1.2;
12909   padding:5px;
12910   margin:0px;
12911   color:#fff;
12912   background-color:#44c;
12913 }
12914 .Pointillism_XY_Remote {
12915   display:block;
12916   vertical-align:middle;
12917   width:20px;
12918   xheight:20px;
12919   font-size:12px;
12920   line-height:1.2;
12921   padding:5px;
12922   color:#fff;
12923   background-color:#4a4;
12924 }
12925 .Pointillism_Canvas {
12926   display:block;
12927   position:relative;
12928   xwidth:20px;
12929   xheight:20px;
12930   xtop:20px;
12931   background-color:#333;
12932 }
12933 </style>
12934 <script>
12935 var points = [];
12936 var replay = [];
12937 var replayx = 0;
12938 function pClearCanvas(can){
12939   ctx = can.getContext('2d');
12940   ctx.clearRect(0,0,can.width,can.height);
12941 }
12942 function Pointillism_1_Clearcanvas(){
12943   pClearCanvas(Pointillism_1_Canvas_1);
12944   pClearCanvas(Pointillism_1_Canvas_2);
12945 }
12946 function PointsReset(){
12947   points = [];
12948   replay = [];
12949   inRepeat = false;
12950   inReplay = false;
12951   Pointillism_1_ClearCanvas();
12952 }
12953 function Pointillism_1_ResetCanvas(){
12954   PointsReset();
12955   if Pointillism_1_Share.checked {
12956     //alert('-- broad cast reset\n');
12957     GJ_BcastMessage('DRAW RESET');
12958   }
12959 }
```



```

13122<!-- Pointillism_WorkCodeSpan -->
13123</span>
13124//<!-- ===== Work -->
13125
13126
13127
13128//<!-- ===== Work { ===== -->
13129<span id="StatCounter_WorkCodeSpan">
13130<details><summary>StatCounter</summary>
13131<h2>StatCounter// 2020-1018 SatoxITS { -->
13132<h2>StatCounter</h2>
13133
13134<div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
13135 (counter as image tag)</div>
13136<style>
13137 .statcounter {
13138   vertical-align:middle;
13139 }
13140 #sc_SatoxITS {
13141   color:#000;
13142   font-size:12pt;
13143   height:30px;
13144   width:100px;
13145   background-color:#ddd;
13146 }
13147</style>
13148<div>
13149<script>
13150   var sc_project=12411639;
13151   var sc_invisible=0;
13152   var sc_security="laeb2a3a";
13153   var sc_https=1;
13154   var sc_jsHost = "https://";
13155</script>
13156<!-- script src="https://statcounter.com/counter/counter.js" -->
13157<script src="https://statcounter.com/counter/counter.js" -->
13158</script> --> (counter by inline script)
13159</div>
13160
13161<input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13162<input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13163<input id="StatCounter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13164
13165<span id="StatCounter_WorkCodeView"></span>
13166<script id="StatCounter_WorkScript">
13167   function StatCounter_OpenWorkCodeView(){
13168     function StatCounter_ShowWorkCode(){
13169       showHtmlCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
13170     }
13171     StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_ShowWorkCode);
13172   }
13173 StatCounter_OpenWorkCodeView(); // should be invoked by an event
13174</script>
13175</div>
13176<!-- ===== Work { ===== -->
13177<div>
13178<!-- ===== Template_WorkCodeSpan -->
13179</div>
13180//<!-- ===== Work -->
13181
13182
13183
13184
13185
13186//<!-- ===== Work { ===== -->
13187<span id="Template_WorkCodeSpan">
13188<details><summary>Work Template</summary>
13189<h2>Template of Work// 2020-0928 SatoxITS { -->
13190<h2>Template of Work</h2>
13191<input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13192<input id="Template_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13193<input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13194<span id="Template_WorkCodeView"></span>
13195<script id="Template_WorkScript">
13196   function Template_OpenWorkCodeView(){
13197     function Template_ShowWorkCode(){
13198       showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
13199     }
13200     Template_WorkCodeViewOpen.addEventListener('click',Template_ShowWorkCode);
13201   }
13202 Template_OpenWorkCodeView(); // should be invoked by an event
13203</script>
13204</div>
13205<!-- ===== WorkCodeSpan -->
13206</div>
13207</div>
13208//<!-- ===== Work -->
13209
13210
13211
13212<!-- ===== Work { ===== -->
13213<span id="OriginalSource_WorkCodeSpan">
13214<details open=><summary>Original Source</summary>
13215<h2>Original Source// 2020-1009 SatoxITS { -->
13216<h2>Original Source of GShell</h2>
13217<input id="OriginalSource_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13218<input id="OriginalSource_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13219<input id="OriginalSource_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13220<span id="OriginalSourceTextElement"></span>
13221<script id="OriginalSource_WorkScript">
13222<script id="OriginalSource_WorkCodeView">
13223   function OriginalSource_OpenWorkCodeView(){
13224     function OriginalSource_ShowWorkCode(){
13225       //showHtmlCode(OriginalSource_WorkCodeView,OriginalSource_WorkCodeSpan);
13226       //OriginalSourceTextElement = OriginalsourceNode;
13227       //console.log('src0=\n'+document.documentElement.outerHTML);
13228       //console.log('src1=\n'+gshell.outerHTML);
13229       showHtmlCode(OriginalSource_WorkCodeView,OriginalSourceNode,
13230         '\n',
13231         '\n',true);
13232     }
13233     OriginalSource_WorkCodeViewOpen.addEventListener('click',OriginalSource_ShowWorkCode);
13234   }
13235   OriginalSourceNode = document.documentElement.cloneNode();
13236   OriginalSourceNode = gshell.cloneNode(true); //=====
13237   //console.log('src0=\n'+document.documentElement.outerHTML);
13238   //console.log('src1=\n'+gshell.outerHTML);
13239   //console.log('src2=\n'+OriginalSourceNode.outerHTML);
13240   OriginalSource_OpenWorkCodeView(); // should be invoked by an event
13241   //showNodeAsHtmlSource(OriginalSource_WorkCodeView,originalSourceNode);
13242   function SaveOriginalNode(){
13243     if( false )
13244       m0 = performance.memory;
13245       mu0 = m0.usedJSHeapSize;
13246       console.log(' heap before clone: '
13247         +m0.usedJSHeapSize+'/'+m0.totalHeapSize);
13248   }
13249   OriginalSourceNode = gshell.cloneNode(true);
13250   if( false )
13251     m1 = performance.memory;
13252     mu1 = m1.usedJSHeapSize;
13253     mu1 = mu1 - mu0;
13254     //alert(' - clone: used heap '+mu0+' -> '+mu1' = '+mu+ ' bytes');
13255     console.log('-- heap aft clone: '
13256       +m1.usedJSHeapSize+'/'+m1.totalHeapSize+'/'+m1.jsHeapSizeLimit);
13257   //OriginalSourceNode = document.documentElement.cloneNode(true);
13258 }
13259
13260
13261 function gsh_SetupPage(){
13262   GShell_Images();
13263   //Indexer after loaded();
13264   //GShell_initKeyCommands();
13265   //GJConsole_initConsole();
13266   GJConsole_initFactory();
13267   GJLink_init();
13268   GShell_init();
13269   GShell_initTopbar();
13270   //VirtualBeaktop_init();
13271   Banner_init();
13272 // Aff_Setup();
13273   Sff_Setup();
13274   window.setInterval>ShowResourceUsage,1000;
13275 //document.addEventListener('keydown',jgshCommand); // should be applied later?
13276 Pointillism_Setup();
13277 FontList_Setup();
13278 showFooter();
13279 GshInsideIconSetup();
13280 SighGlass_Setup();
13281 spawnAckmonGo();
13282
13283function OnLoad(){
```

```
13284     SaveOriginalNode();
13285     Gsh_SetupPage();
13286 }
13287 document.addEventListener('load',Gsh_SetupPage);
13288</script>
13289<details>
13290<!-- OriginalSource_WorkCodeSpan } -->
13291/* //</span>
13292//<i-- ===== Work } ===== -->
13293
13294
13295
13296//</div>
13297//<br><script>OnLoad();</script></span>
13298
```